FEDERAL STUDENT AID

We Help Put America Through School

FSA Modernization Partner

NSLDS II Reengineering
Data Architecture Detailed Design

Version 2.1

November 26, 2002

Table of Contents

Document Control

Version Number	Description	Release Date	Author
1.0	Initial Design of DDL, Data Model, and NSLDS to NSLDS II EDW database	09/30/2002	Terry Helwig
1.1	Updated to add details on development database and AIX environment on IBMP1	10/15/2002	Bob Harbus
1.2	Based on user comments, took out section on Informatica managing metadata layers	10/21/2002	Terry Helwig
1.3	Update structure to reflect Data Mart and replication changes.	10/29/2002	Terry Helwig
2.0	Added sections on security, capacity, and replication.	11/06/2002	Terry Helwig
2.1	Updated NLSDs Assumptions to include reporting requiremens	11/14/2002	Terry Helwig

1 Introduction

The National Student Loan Data System (NSLDS) was established as part of the Higher Education Act of 1965, as amended, to provide a comprehensive repository of information about Title IV recipients and their loans, grants, lenders, guaranty agencies (GAs), servicers and schools. As NSLDS has evolved since its implementation in 1994, it has been hampered by a number of challenges due to discrepancies between the quality and timeliness of NSLDS data and the system of record, its analytical capabilities, and operating costs. Given these challenges, a project to modernize the system has been undertaken with the following objectives:

- **Data Warehousing:** Improve usability of the NSLDS data repository through new tools.
- **Internal Federal Student Aid (FSA) Direct Access:** Improve customer satisfaction through better quality and usability of NSLDS information.
- **Outsourced Enrollment Tracking:** Balance FSA data needs with burdens placed on the financial aid community.
- **Financial Partner Data Feed Reengineering:** Take greater advantage of data resources available within FSA and from the financial aid community
- **Common Record Extension:** Improved financial integrity, reduce FSA costs associated with NSLDS and related operations.

The first phase of the NSLDS Reengineering effort is called NSLDS II. The first release of NSLDS II will focus on re-platforming the existing system to a new modernized architecture. The existing NSLDS data will be transformed to a data warehouse architecture with a dimensional data mart (based on reporting requirements). Additional, this phase of NSLDS II will focus on the Internal FSA Direct Access Opportunities as well as assessing ways to support existing requirements through the NSLDS II or other modernized systems. Later releases of work will address the other objectives and additional requirements that FSA may have.

2 Scope

This document details the data architecture design for NSLDS II, which includes the following design areas for the Enterprise Data Warehouse (EDW) and Data Mart databases:

- **Data Definition Language (DDL):** The DDL is used to create and delete databases and database objects. Database administrators use the DDL to create new and empty tables as well as delete and modify existing tables.

- **Logical Data Model:** A logical data model is a precise and unambiguous expression of business facts. The logical data model shows entities, attributes, keys and relationships.

- **Data Mappings from the EDW database to the Data Mart database:** This section details the replication and transformation of data from the NSLDS II EDW database to the NSLDS II Data Mart database.

The subsequent issues are deemed out of scope of this document as they were discovered during the later stags of the detail design phase or have been omitted from this version due to a lack of source documentation from the legacy system vendor:

- **IBM 3494 Automated Tape Library (ATL) data server and Magstar Virtual Tape Server (VTS):** During the detailed design phase, it was disclosed that an Automated Tape Library existed that provided real time access to over roughly 25,682 - 50 GB tapes or roughly 1,284 TB of storage. As part of the technical reassessment effort to apply NSLDS II requirements to all technical platforms (mainframe as well as mid-range), the Automated Tape Library architecture will be examined.

- **Trading Partner Tape Loaders:** Currently, NSLDS trading partners (GAs, Schools, Servicers, and the debt collection system) send in data using three different tape interfaces (3490, 3480, and 3420 IBM tape servers). Due to the age of the 3480 and the 3420 tape servers, it is anticipated that these devices cannot be integrated into the new mid-range platform. The technical reassessment phase of the NSLDS II project will address this issue and propose alternative solutions.

- **Archive Strategy:** There is currently no archive requirement for the NSLDS system. The NSLDS II data architecture team is currently reviewing a variety of archiving strategies with FSA's data management team, however at this time no archiving approach has been approved. As FSA refines its archiving policies, the NSLDS II data architecture team will incorporate new requirements into future releases.

- **Statistically Abstract Database (STAB):** The STAB database represents a subset of the NSLDS data that is statistically accurate for business users to test queries, perform complex statistical analysis such as trending reports and forecasting, and to allow auditors the ability to validate samples of data. The NSLDS II data architecture team has proposed that the STAB database should not be converted to NSLDS II as there are very few actual users of the database. Additionally, it is expected that there will be a performance increase of the EDW and Data Mart databases that should allow better performance of queries. The

MicroStrategy reporting tool will meet the reporting requirements of the business users upon request.

3 Assumptions

For the current design of the NSLDS II data architecture, the following assumptions and requirements have been incorporated:

- No operational data store or staging area is required in order for data to be loaded into NSLDS II. This is due to the fact that data from trading partners is loaded directly into the data warehouse with no consolidation or accumulation of files required.

- The legacy NSLDS is the data store for the Financial Aid Professional (FAP) website and the Student Access Financial Review website. NSLDS II will continue to provide this functionality.

- All reporting requirements currently in the NSLDS system will be met by the NSLDS II system. Additionally, any reporting requirements (adhoc or standard) that need to use the 3rd normal form database will be met by utilizing queries against the EDW database in the NSLDS II system.

- The reporting requirements necessitate that reports must be able to be generated with real time information including updates made on the website.

- The NSLDS II data architecture will leverage the existing NSLDS database design in order to leverage its 3rd normal form structure.

4 Approach

Based on the assumptions and requirements outlined above, the NSLDS II data architecture has been broken down into five main components:

- **Enterprise Data Warehouse (EDW):** The EDW will be designed as a close approximation of the legacy NSLDS database. This approach will allow the EDW to leverage the existing logic for loading data, procedural logic for calculations, and update logic of the FAP and Student websites.

- **Extract, Transformation, and Load (ETL) Process:** This process details the loading of data from the trading partners along with the replication of data from the Enterprise Data Warehouse database to the Data Mart database.

- **Data Mart:** This database contains a subset of the Enterprise Data Warehouse designed for aggregation of data for analytic reporting needs.

- **Metadata Repository:** The Metadata repositories are "data about data" that define the business and technical needs of the users through natural language rather than relational database terms.

- **Client Access:** There are multiple clients to NSLDS II that access data through websites, data feeds, and direct SQL access. The data architecture has to be flexible enough to handle all of the client user requirements.

4.1 Enterprise Data Warehouse (EDW)

Figure A represents a graphical depiction of the NSLDS II data architecture. This diagram is subject to change based upon technical limitations discovered during the build cycle. The current design is for the EDW to be populated through the conversion of the NSLDS mainframe database (please reference the NSLDS II Reengineering Data Conversion Detail Design documentation for details.) This EDW database will approximate the current NSLDS design (a 3rd normal form structure) and serve as the operational database for NSLDS II. The design of the EDW in 3rd normal form is the result of the following requirements:

- A transactional repository for the Financial Aid Professional (FAP) website and the Student Access Financial Aid Review website.

- Account for existing NSLDS procedures such as the cohort default rate (CDR), Account Maintenance Fee (AMF), Payment Reasonability, Federal Receivables, Student Transfer Monitoring, Pre-Screen, Post-Screen, and Loan Processing and Issuance Fee (LPIF). To do this, simulating the existing legacy 3rd normal form database structure will allow a decrease in SQL calculation development time and maximize logic re-use.

- Decrease in load times for trading partners to and from the NSLDS II system. Part of this requirement is met by re-using the existing structure of the legacy NSLDS database and logic for the loads. (In addition, performance will be realized by using 3rd party consumer

off the shelf tools such as IBM MQSeries and Informatica to provide data transport services, extract, transform, and load processes).

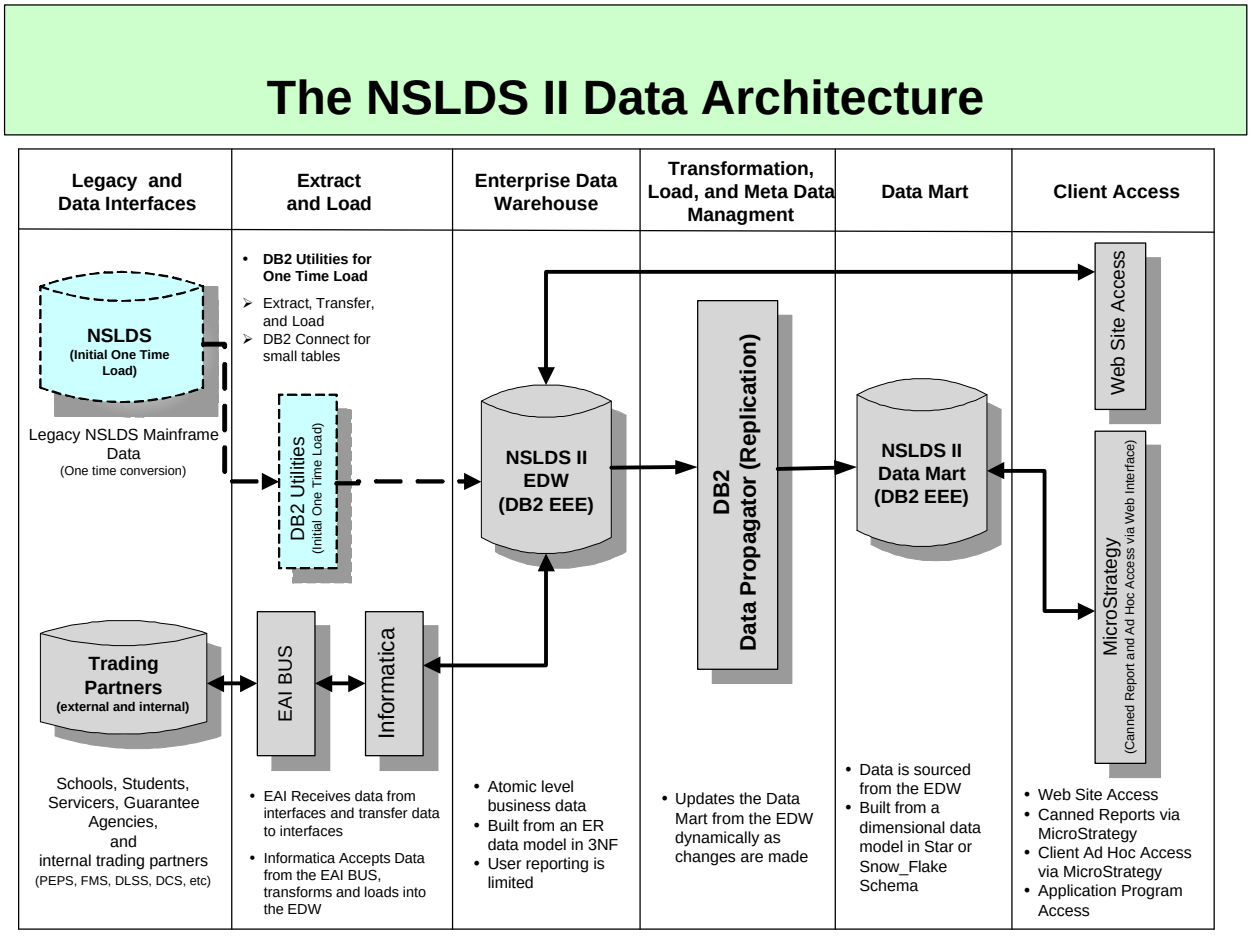


Figure A. NSLDS II Data Architecture

4.2 Extract, Transformation, and Load (ETL) Process

Two COTS tools, IBM's MQSeries and Informatica, will be used to load data into NSLDS II and to perform all transformation and replication activities. These COTS tools provide the following benefits:

- Lower volume of code to maintain.
- Less development time by leveraging existing code or GUI interfaces for data loading, transformations and replications.
- Less maintenance and operation cost as vendors continually update their products with new features and fixes.

The Enterprise Application Integration (EAI) team provides a set of common technology services that enable the sharing of processes and data of disparate systems to support end-to-end business processes across FSA. The EAI architecture enables many "stovepipe" applications to exchange information using common, reusable methods and infrastructure. By leveraging IBM's MQSeries software suite, the EAI team provides the capability to integrate web-based applications, Data Warehouse environments, COTS packages, and existing legacy systems

within the FSA technical environment. The cost of the software suite and hardware to run it is shared among the projects within FSA that utilize the services.

Leveraging FSA shared resources; NSLDS II will use the EAI bus to dynamically load data via the MQSeries interface from trading partners who are on the bus and the Student Aid Internet Gateway (SAIG). All data will be delivered via EAI to directories on the Informatica server, which will perform the loads, edits, and any transformations into the NSLDS II EDW database.

4.3 Data Mart

The Data Mart database will provide reporting capabilities for NSLDS II. It contains a subset of the EDW for specific ad hoc queries and standard reports. This database is a multi-dimensional database with the ability to drill down to a specific level of detailed defined by the user community. The table structure is such that a single update on the EDW database may result in updating multiple tables in the star schema of the Data Mart database. This database uses the COTS tool, MicroStrategy, for its reporting presentation layer. The MicroStrategy tool is the FSA standard reporting application for on-line analytical processing (OLAP). It is a web based tool that analyzes data from a number of different perspectives or business dimensions by drilling up or down through the data provided in the Data Mart. (Please refer to the NSLDS II Reengineering MicroStrategy Project detailed design document).

4.4 Replication

One of the requirements from the reporting community is that updates made to NSLDS II through the FAP website are available immediately through NSLDS II reports. To meet this requirement, different replication architectures have been examined to ensure that data in the NSLDS II EDW database is replicated in real time to the reporting Data Mart database. After initial reviews of third party products (Informatica, Business Objects, IBM MQ Series), IBM's Data Propagator replication tool was selected based on ease of use and best performance. This tool uses a graphical user interface to map and transform data from the 3rd normal form EDW database to its corresponding location in the dimensional Data Mart database. An example of this replication and transformation is provided later in this document.

4.5 Metadata Repositories

The Metadata repositories are "data about data" that define the business and technical needs of the users through natural language. Objectives of the metadata are the following:

- To provide a means to improve the productivity of the administrators / developers and the reliability of the business intelligence solution.

- To provide a means to assist business analysts in locating and understanding the data in the data warehouse environment.

There are two main areas that contain metadata in the NSLDS II data architecture. The MicroStrategy reporting application creates a metadata layer with pointers to the actual data, attributes of the data, and parameters of the reporting project. Additionally, the Informatica application has a metadata layer that stores configuration information and data about the different transformations performed and data loaded into the EDW database.

4.6 Client Access

Client access to the application will be primarily through the NSLDS II Financial Aid Professional (FAP) website and the Student Access Financial Aid Review (SAFAR) website. These websites are for schools, lenders, guaranty agencies, and students to examine the history of various loans. Additionally, the FAP website will have a link to the MicroStrategy reporting server where users will be able to run ad hoc and pre-built reports. The websites will access the EDW database for updating information and for read only information on the website. A secondary data access tool will be provided for a small super user community that needs access to the EDW and Data Mart databases for running highly customized ad hoc queries.

5 NSLDS II Data Definition Language (DDL)

The NSLDS II DDL will be used to create the tables inside the enterprise data warehouse and Data Mart. The DDL includes table names, column names, data types as well as partitioning keys and indexes. Using pre-packaged modeling software, Computer Associates AllFusion ERwin Data Modeling suite, a DBA will create the DDL. Additionally, the ERwin software will be used to help create and maintain all of the databases, data warehouses and enterprise data models. By using this software databases can be developed more quickly with more efficiency and maintainability.

ERwin is used to reverse-engineer the existing NSLDS database and create the initial copy of the DDL. The DBA will then modify the file to reflect the new requirements and architecture of the NSLDS II data warehouse environment. The DBA will need to modify the DDL to implement an indexing strategy and choice of singular or composite partitioning keys for multiple-partitions tables.

There are two separate sets of DDL files defined for NSLDS II. The first will be used to construct the EDW portion of the system. The second will construct the tables and columns that comprise the NSLDS II Data Mart.

The following examples of syntax provider for a better understanding of what the DDL does.

- The *CREATE DATABASE* NSLDS command will create an empty database called 'NSLDS'.

- The following *CREATE TABLE* statement adds an entity called STUDENT in the database with the field's *first name*, *last name* and *student ID*. When a row is added to this table, it must contain a value in each of these columns, hence the NOT NULL constants. The final clause contains the partitioning key. The partitioning key is used in a multiple partition DB2 UDB EEE environment to tell DB2 UDB which column values to use in a hashing algorithm to determine placement of new data rows and locate existing data rows.

```
CREATE TABLE student
(first_name     char(20) NOT NULL,
last_name       char(20) NOT NULL,
student_id      INT    NOT NULL
PARTITIONING KEY
(student_id))
```

Please refer to *Appendix A Data Definition Language for NSLDS II EDW and Data Mart* for the complete data definition language of the EDW and the Data Mart. It is important to note that many database privileges and roles must be assigned to the user executing these commands in order for them to be executed correctly.

6 Logical Data Model

The new NSLDS II data architecture will consist of two logical data models, the EDW and the Data Mart. Data Models contain entities, attributes and the relationships between entities. Entities are items of relevance about which information can be kept. They refer to persons, places, things or concepts. Examples of entities are Loans, Students, Schools, and Lenders. Attributes are quantitative or descriptive characteristics of the entity. For instance, student name is a descriptor detail that users associate with the Student entity. Keys identify properties of entities, and relationships show association between two entities. This information, the entities and relationships between them, is represented in the logical data model in Entity / Relationship (E/R) diagrams. The NSLDS II logical data model is recorded in the ERwin data modeling software. The EDW and the Data Mart data models will be presented in *Appendix B*.

6.1 Purpose of Logical Data Modeling

The two primary reasons for creating a logical data model are:

- **Improve communication:** Communication between the business users of the data and the developers is essential to a successful development project. Eventually, all of the work that a business does comes to rest in its data. Understanding the business nuances that are inherent in the data is one of the quickest and surest ways for a developer to obtain an understanding of the business. A logical model helps depict those nuances as it denotes entities and their relationships to each other. When developing a data warehouse, the developer needs this understanding to create a data warehouse that can answer the questions posed by the business.

- **Provide input into physical design:** The logical data model is one of the inputs into the physical data design. The other main inputs are access patterns, performance criteria, and technical infrastructure. These inputs will be documented in future releases of this document.

6.2 EDW Data Model Components

The EDW data model will have the similar logical and physical structures as the legacy NSLDS data model. The underlying EDW data model is in the 3rd normal form (3rd normal form is defined as every non-key attribute should be functionally dependent on the full key and nothing but the full key). This also means that the EDW data model has been normalized to ensure consistency, reduce redundancy, and maximize stability in the model. Normalization drives the following:

- Ensures all entities or attributes occur only once in the model.
- Assists in associating data attributes with entities based on properties of the data rather than on application requirements.
- Puts the keys and data together so that it can be maintained without jeopardizing information integrity.

The EDW data model is created on the ERwin data modeling tool by reverse engineering the existing NSLDS DB2 database on the mainframe. All NSLDS tables were reverse engineered into ERwin and only entities required for NSLDS II are selected into the EDW data model. The EDW data model represents all the entities, attributes and relationships in a single E/R diagram for all subject areas.

6.2.1 EDW Subject Area Diagrams

A total data model diagram that lists all inter-relationships is too large for viewing on a normal 8.5 x 11 page. For this deliverable, the E/R diagrams were created by subject area for easier for viewing. A subject area centers on a particular subject matter as opposed to being generic and open to many different types of information.

The following are the major subject areas identified within NSLDS II. Each are displayed in *Appendix B*:

LOANS
STUDENTS
SCHOOLS
LENDERS
GUARANTY AGENCIES
DEFAULT RATES
FDSLP
TRANSFER MONITORING

For each subject area listed above, an E/R diagram has been created in ERwin.

6.2.2 EDW Enhancements

While the EDW data model has similar data structures as the legacy NSLDS mainframe database, a few changes that have been applied. The first is that each EDW table will be expanded to include a date and time stamp to signify the occurrence of an update event in each row in the table. This addition is for administrative purposes only and is not meant as a design change to any business logic. If a problem in the database occurs, this date time stamp will allow administrators to trace a problem to the time of occurrence on the database. Additionally, a date time stamp may be used in the future for rollback, error resolution, purposes. This new TIMESTAMP column is added to each table to reflect the date and time an update is applied to each row in the table. The new column will be as follows:

DT_TM_STAMP TIMESTAMP NOT NULL WITH DEFAULT

If this column of type TIMESTAMP was added to an existing table, the default for the timestamp will be as shown below when the table row is accessed.:

0001-01-01 00:00:00.000000 where

Year = 0001

Month = 01
Day = 01
Hour = 00
Min = 00
Sec = 00.000000

For the EDW the TIMESTAMP column DT_TM_STAMP will be part of the initial table creation and as such the column value either be supplied when a row is inserted into the database table, or if not supplied, the CURRENT TIMESTAMP at time of insert will be used to populate the column.

When updates are made to the EDW on a daily, weekly and monthly basis, each update will provide a current timestamp in the DT_TM_STAMP column. The ETL tool will examine this column on each update to the EDW.

Other changes made to the EDW entities and attribute names are as follows:

NSLDS Entity	NSLDS Attribute	NSLDS II EDW Entity Name	NSLDS II EDW Attribute Name
FS_SBMTL_RUN_ERR	FFEL_DUP_ID	FS_SBMTL_RUN_ERR	IND_SEP_LOAN
GA_SBMTL_RUN_ERR	FFEL_DUP_ID	GA_SBMTL_RUN_ERR	IND_SEP_LOAN
LOAN	FFEL_DUP_ID	LOAN	IND_SEP_LOAN
LOAN_RPMT_PLAN	DT_ENTR_RPMT	LOAN_RPMT_PLAN	DT_ENTR_RPMT_PLN
STU_DEM	CALNDR_YR	STU_DEM	AWARD YR
SCH_BR_SVR		PERK_SVR	
LEN_BR_HOL	LEN_BR_CODE	LEN_HOL	
LEN_BR_HOL_SVR		LEN_HOL_SVR	
LEN_BR_SVR	LEN_BR_CODE	LEN_SVR	
STU_BR_ID		STU_DESIG	
SCH_SBMTL_RUN_ERR		PERK_SBMTL_RUN_ERR	
SCH_SBMTL_HIS		PERK_SBMTL_HIS	

Figure B. EDW Database Changes

6.3 Data Mart Data Model Components

In contrast to the EDW database's 3rd normal form design, the Data Mart database uses a dimensional model (DM) design that seeks to present the data in a intuitive framework that allows for high-performance access. It is very different from traditional entity-relation modeling often used in 3rd normal form databases by implementing data in a star or snowflake schema in order to drilling down from high level to detail level data. Traditionally, a dimensional model database is composed of one table with a multipart key called the fact table and a set of smaller tables called dimension tables. Each dimension table has a single-part primary key that corresponds exactly to one of the components of the multipart key in the fact table (Please see NSLDS II Reengineering Reports Detailed Design for a full discussion of the data model and its design). The Data Mart data model is created with the ERwin data-modeling tool and broken out into logical subject areas.

6.3.1 Data Mart Subject Area Diagrams

Based on ad hoc reporting needs, jointly created by business user groups, NSLDS II team, pre-existing standard reports, and administrative requirements, the Data Mart database is segmented logically with the following major subject areas:

- CDR
- Date
- GA Sum
- Grant
- Guaranty Agency
- Lender
- Loan
- Misc. Lookups
- Organizational Contact
- Prescreening
- School
- Servicer
- SSCR
- Student
- Transfer Monitor

For each subject area listed above, a relationship diagram has been created in ERwin (Please see *Appendix B EDW and Data Mart Data Models*). Additionally, this design document details the mappings from the EDW source tables to Data Mart destination tables in *Appendix C Data Mappings* from NSLDS II EDW to Data Mart. These mappings will be used by the replication process to populate the Data Mart database dynamically, or in batch, resulting from the updates made to the EDW by the trading partners or websites.

7 Replication from EDW to Data Mart

The NSLDS II data architecture is designed as a 3rd normal form enterprise data warehouse. This database acts as a staging area for interface loads, website repository updates to loan level data, and a transactional database for procedural calculations such as the Cohort Default Rate (CDR), Account Maintenance Fee (AMF), Payment Reasonability, Federal Receivables, Student Transfer Monitoring, Pre-Screen, Post-Screen, and Loan Processing and Issuance Fee (LPIF).

The Data Mart database design has two major requirements, to increase the performance of the business user reporting requirements and to remain current with any updates being made from the FAP website or batch feeds. To meet these requirements, the Data Mart is built in a dimensional format for consolidation of reporting information and the ability to drill down to lower level loan detail. Additionally, it is replicated dynamically with the EDW via the DB2 tool Data Propagator. Its size approximates the size of the EDW (see *Appendix F Capacity Planning for NSLDS II*). DB2 Data Propagator consists of the following components:

- A CAPTURE module that reads changes from the DB2 log files (on the EDW database) and inserts those changes into staging tables or Change Data (CD) tables (on the Data Mart database).

- Once the data has been captured, the APPLY module applies the data from the change data tables to the target system (Data Mart). During the apply process transformations can be made to the data (going from a 3rd normal database to a dimensional model).

For the purpose of explaining IBM's Data Propagator replication application, this document will use the examples of the NSLDS II EDW and Data Mart databases.

7.1 Data Propagator's CAPTURE Component

When changes are made to a table on the source database (the EDW database in this example), DB2 writes log records to another table. These log records are used for database recovery and for replication. Each source table on the EDW has a corresponding Change Data (CD) table on the Data Mart where the captured changes are stored. The CD tables are created when the Database Administrator (DBA) defines the replication source tables. The DBA can also choose to capture only a subset of the source table columns if needed or capture the values before the change is made, and /or the values after the change is made. The log record sequence number of a change is used to uniquely identify that change. DB2 CAPTURE holds the changes in memory until a COMMIT is issued for those changes. When a COMMIT is issued for a transaction that involves a replication source table, CAPTURE inserts the changes into the appropriate CD tables and stores the COMMIT information in the Unit of Work (UOW) control table. When CAPTURE detects a ROLLBACK (backing out a change), it removes the associated changes from memory.

7.2 Data Propagator's APPLY Component

In the NSLDS II data architecture design, the Captured changes are applied to target tables on the Data Mart database by Data Propagator's APPLY programs (see Figure C: EDW to Data Mart replication). The Apply program can run on any server and must have connectivity to

both the source and the target servers. Data can be filtered by column, by row, joined with other data (using views) or transformed with SQL expressions during the APPLY process. The changes from CD tables are applied for each table separately in the same order they occurred at the source. The Apply process can be configured as a batch process or a process that runs all the time.

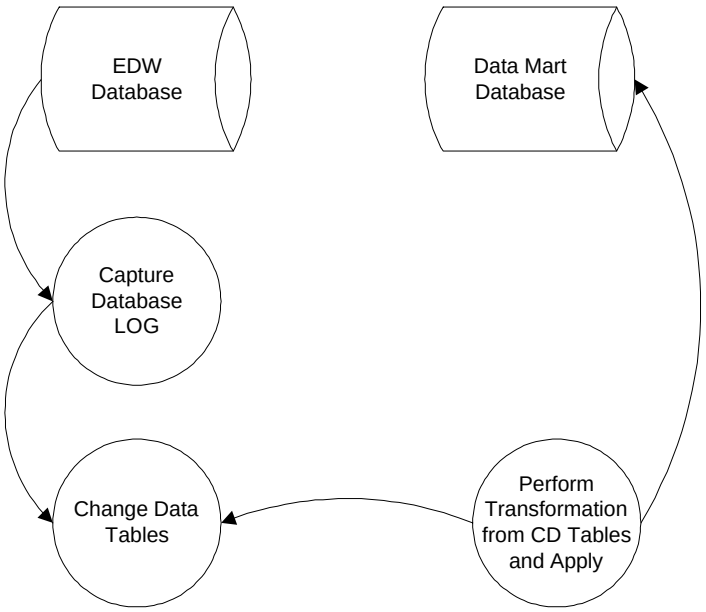


Figure C. EDW to Data Mart Replication

7.3 Replication Center

To control the Capture, Apply and transformation of data, Data Propagator has anther tool called the Replication Center, which is a graphical tool for administration of replication between the EDW and the Data Mart. The replication sources, target and the source-target maps are defined using this tool. The replication components will be configured such that, as soon as an application (web or interface) updates any of the EDW tables, the CAPTURE program will read the entry from the DB2 logs on the EDW and store the information in the staging area of the Data Mart. The APPLY program will then configured using a pre-determine interval or on commit to update the Data Mart database with the EDW changes. Additionally, the APPLY program will perform the desired transformation from the 3rd form normalized EDW database to the dimensional Data Mart schema before applying the changes.

7.3.1 Replication Example: Student Table

The following is an example of some of the activities that a DBA replication administrator would do in setting up replication between the EDW and the Data Mart for the NSLDS II project.

- The first step for the DBA is to configure connectivity to source and target databases. Replication center will initially create the required replication control tables in the source database (EDW).

- The DBA will next register the source tables in Replication Center. The program is flexible enough to allow the DBA to select any of the tables from source database. In this case, the DBA defines the Student table from EDW as source with the “change capture” option (the other option for this feature is full refresh).

- When the Student table is defined as a source, the table is altered with the definition ”Data Capture Changes” attribute. This enables full-row logging for the source table and allows Capture to set the before and after images of the changes to the Student table. Additionally, a row is inserted in the Capture Control system table indicating the registration of a source

- At this point, CD table is created to hold the captured changes to the EDW. This table has columns that were selected while registering the source.

- The next step is to define a “Subscription Set”, which is a mapping from source to target table. Each mapping is called “Subscription Members”. While creating a subscription set for the Student table, the DBA will need to specify the control server (same as the source server), capture control server (same as source server) , target server and scheduling information. The schedule can be defined as time interval or an occurrence of any events.

- The final step will be to define the “Subscription Member”, which will define the source and target column mappings with all transformations. At this point, the DBA would start the “Capture” and “Apply” programs and test the data replication configuration.

8 NSLDS II Capacity Estimate

During the capacity planning effort of the NSLDS II detailed design, Raytheon provided capacity data related to the current NSLDS database. This data was extrapolated using the requirements of the NSLDS II system. Multiple views of the data are presented (see *Appendix NSLDS II Capacity Planning*).

- The existing mainframe NSLDS Production Environment size compressed in Terabytes.
- Estimation of the NSLDS II Mid-Range Production Environment size uncompressed in Terabytes.
- Estimation of a NSLDS II Mainframe Environment size compressed in Terabytes (production, test, and development).
- Estimation of the NLSDS II Mid-Range Environment size uncompressed in Terabytes (production, test, and development).

Note that these numbers are estimates. A review process during the initial phases of the technical reassessment phase of the NSLDS II project will focus on validating these estimates.

8.1 Capacity Assumptions

The current capacity estimates for the NSLDS II design were made with the following assumptions:

- The NSLDS II Enterprise Data Warehouse is a close approximation of the existing NSLDS database.
- The NSLDS II Data Mart contains roughly 80% of the same tables as the EDW and will approximate the size of the EDW.
- Due to compression and smaller sample sizes, the mainframe database environments storage requirements are a fraction of the mid-range environments.
- Raytheon has verified that there is an automatic tape library with roughly 25,682 live tapes on NSLP. With each tape equal to 50GB, this represents roughly 1,284 TB as the tapes neither deleted nor in scratch status.
- Raytheon has verified that the automatic tape library is growing at approximately the same rate as the NSLDS database.
- Raytheon has verified that of the 1 TB of database space allocated for NSLDS production 45% of this is compressed (or approximately 1 TB / 1 - .45 = 1.81 TB of disk space will be needed for NSLDS II production if compression is not used).
- Raytheon has verified that there is roughly 120GB of data is written to the database each year. Of this data, a large portion of it is loaded from the trading partners via tape and

electronic format with only 5% of the traffic due to the website. A conservative estimate has been calculated that roughly 25mb a day are written to the website (120,000 MB / 365 days = 328.76 MB, 5% = 16 MB a day. To be conservative and account for peaks, 25 MB will be assumed).

- The NSLDS II team is assuming that once the system is rolled out, there will be a 30% increase in capacity due to increased web usage and additional requirements being added to the system.

- The NSLDS II team is assuming that for a mid-range solution the Test2 environment will be able to replicate at least two full production environments for training, system testing, user acceptance testing, and performance testing activities.

- The growth rate of the development and test environments will flatten out over time as the system matures and less space is needed.

- As only a subset of data will be needed, the development environment will be sized as half of the initial production environment.

- Since the Test 1 environment will be for system testing, it is assumed it will be equivalent to the development environment.

- The growth rate for Development and Test1 environments will assume to be 10% to adjust for new requirements.

- The Test2 environment will be able to replicate at least two full production environments for performance testing, regression testing, and system testing activities. It will be twice the size of production.

8.2 Capacity Planning Results

Details on capacity planning are located in *Appendix F Capacity Planning for NSLDS II.* A high level summary of these details is compiled in the figure *NSLDS Capacity Planning* below. From this table it is interesting to note the difference between the new NSLDS II Mainframe Capacity data and the new NSLDS II Mid-Range capacity data for all environments (In 2003, these numbers were 11.60 TB and 17.82 TB respectively). This is due in part to the compression of the NSLDS II data on the mainframe with the consolidation of interface files and procedural files. It should also be noted that the capacity estimates may change based on modification of the above assumptions or as a result of the review by FSA, CSC, Raytheon, IBM, and Accenture to be performed during the next phase of work.

Year	Production Environment ONLY		ALL Environments (Production, UA, Testing, Development)	
	Total NSLDS MAINFRAME Compressed (TB)	Estimated NLSDS II MID_RANGE (TB)	Estimated NLSDS II MAINFRAME Compressed (TB)	Estimated NSLDS II MID_RANGE (TB)
2003	1.95	4.45	11.60	17.82
2004	2.24	5.72	13.20	20.42
2005	2.58	6.75	15.01	22.92
2006	2.97	8.77	18.47	24.94
2007	3.41	11.41	22.95	27.58

Figure D. NSLDS II Capacity Planning

9 NSLDS II Security Approach

9.1 Introduction

This section will provide an overview of the existing security for the NSLDS legacy system and provide an example of the existing method for user authentication. It will than examine the high level approach for migrating security information from the legacy NSLDS to NSLDS II, the security architecture of the NSLDS II re-platformed websites (FAP and Student), MicroStrategy reporting application, EAI MQ Series bus, and Informatica.

9.2 Assumptions

This is not a detailed design of NSLDS II security. Rather it is a high level approach that will be used to create a statement of work for a separate effort to develop a detailed security approach and plan that will cover include all aspects of security (hardware, network, software, facility, and personnel) for review by the FSA security officer.

9.3 Existing Legacy NSLDS System

The existing NSLDS security is based on the function that a user is planning to perform and by the means used to access information from the database. NSLDS has three major security layers, they are:

- **User Identification and Authentication**—The RACF User ID and password security, or PIN, depending on which site is being accessed, are used to identify that the user has access to the NSLDS mainframe and is authorized to execute server transactions from the Web application running on the mainframe.

- **Application Access Security**—Database tables are used to verify that the user is authorized to run the application from the Web, and to establish the Web pages that the user is authorized to visit.

- **Data Transmission**—Data is encrypted using Secure Sockets Layer (SSL) and server certificates.

To protect the confidentiality of information, and to prevent the unauthorized tampering of NSLDS data or the mainframe itself, the following security measures have been instituted:

- Restrict access to NSLDS information to users who have been provided access by FSA and have a valid userID in RACF and the internal NSLDS tables.

- Access by such authorized users is limited to transactions created specifically for the distinct RACF user groups that access or update information in only those NSLDS tables containing relevant aid information.

- Access is limited to specific aid information requested using on a borrower's Social Security Number and other identifiers.

9.3.1 User Identification and Authentication

9.3.1.1 RACF Security

The Resource Access Control Facility (RACF) software determines the user's identity based on a seven-character (or shorter) user id (UID), which is stored in a RACF flat file on the NSLDS mainframe. The UID is defined with the types of access privileges that the user possesses. The privilege portion of the UID database defines the functions that the UID can perform (e.g. CICS access) and access to individual transactions. Also included are fields to identify the user as they are using the system and which functions they can perform (e.g. security administrator, system auditor, etc). Authentication is based on matching a personal secret password to an encrypted password in the RACF file (the password is stored one-way encrypted non-readable).

When the UID is successfully authenticated with the password, access to the system is allowed. New users can submit the UID request form and follow the defined procedures to obtain access to the NSLDS system.

9.3.1.2 User Security Groups

There are three main user security groups in relation to website access, ad hoc reporting access, and batch user access.

9.3.1.2.1 Web Groups:

There are two web groups associated with their respective websites. The student group accesses the NSLDS Student Financial Aid Web Site while the financial aid partner group accesses the NSLDS Financial Aid Partner website.

9.3.1.2.1.1 Student Group:

- Student Access Financial Aid Web Site utilizes the third party NCS Pearson's PIN website. When a user from the student group accesses the student site for logon, they are a re-directed to the NCS server PIN website at the Virtual Data Center (VDC) in Meriden, Ct. This is a secure site behind VDC firewalls that is used by several of the FSA sites for pin validation. On the NCS pin site, the user fills out their SSN, date of birth, first two letters of last name, and the FSA PIN. The NCS site authenticates the information, then an accepted or rejected message that is encrypted using the FSA standard encryption package (BSAFE built by RSA Security, Inc) and is sent back to the Student site where it is decrypted and processed.

9.3.1.2.1.2 Financial Aid Partners Group:

Members of the Financial Aid Partners Group (e.g. FAP, Ed users) access the NSLDS website via the logon.asp page. This page creates a visual basic (VB) object and associated attributes that call a COOL:gen procedure (i.e. "Action block") named WSO1_RACF_LOGON_SVR. First a generic "WEBSA" client userid and "WEBSA01" password are passed as attributes to the action

block. This password and UserID allow for DB connectivity. The ID and password that the user entered are also assigned as attributes to the VB object and imported into the action block. This action block in turn calls an external Cobol action block named WS01E01_WEB_EAB_LOGIN_MAINT (and again exports the entered UserID and password to this external program). This Cobol code calls the RACF subroutine that checks the RACF flat file and verifies that the user has access rights to the website.

9.3.1.2.2 Ad Hoc Report Group:

Members of the $ED and $EDNSLBR function groups, (comprised of ED users) use RACF OS security to perform ad hoc queries against roughly 100 tables within the database. These users have been granted access to table views.

9.3.1.2.3 Batch User Group:

- The CA7 Batch Scheduler (like MCRON or CRON schedulers on UNIX) uses BATUSER authority to execute batch jobs that are monitored by the CSC scheduling team.

9.3.2 Application Access Security:

After the user has successfully logged in, web application security is handled through ASP code and procedure calls to tables in the database, (i.e. Location Group, Function Group, Web Group, etc.) which is termed application level security.

The Web Page, Location Group, and Function Group tables are used in the same manner as is currently done in the CICS online system to perform application security. Once the user has signed on with his or her User ID and password, these tables are read to determine the Web pages that the user has access to and to create a dynamic menu of these locations. Each time the user attempts to access one of the pages on the site via a link or button, the user's clearance is confirmed via these tables before the page will open up for the user. Once an authorized user has accessed a page, that user may submit requests for a student's financial aid information using the same identifiers required on the corresponding CICS screens.

9.3.3 Data Transmission:

9.3.3.1 Server Certificate:

- To ensure that the connection to the user is secure, a Web server certificate is used to authenticate and secure the session connection for that user's session. Sessions without activity will be timed out after 10 minutes. If the user enters their User ID and password correctly, navigation will flow to the Privacy Act page. The user must accept the terms of the Privacy Act to continue. All confidential information is passed in an encrypted format. While the system accepts 40-bit as a minimum encryption level, if the user has a domestic browser that supports 128-bit encryption, the encryption will be at that level. The encrypted

data is decrypted and passed along to the NSLDS mainframe where a transaction is executed that retrieves aid information for that student. This retrieved data is dynamically written as a Web page to be viewed by the user.

9.3.3.2 Client Security:

NSLDS has provided instructions for FAP website users for configuring various browser settings. The caching of secured pages to disk is one of them. If the user is using Netscape 4.x, encrypted pages are not cached to disk, so there is no setting for this. Encrypted pages are also not cached to disk with Internet Explorer 4.x–5.0. Autocomplete for logon forms is another. If the parameter is set to "off", it prevents saving logon IDs and passwords on the user's workstation. Additionally, in instances where a called page is linked to by a calling page that needs to provide information that is sensitive or should be restricted from the user's view, the information will be communicated through hidden inputs, posted forms or session variables.

9.3.4 Server Administration:

- The midrange group at the Virtual Data Center (VDC) for the NSLDS FAP website maintains the development and production servers. The NSLDS FAP site can only be administered and maintained by this group and the Raytheon developers that are responsible for this site. Access to administration functions is secured by Microsoft NT Server security, and requires a User ID and password. No NSLDS data is maintained on the server's hard drive.

Microsoft NT Server security also provides a logging feature to monitor and document security breaches. Other than standard security reviews by the midrange group at the VDC, no special security review procedures have been established. However, random penetration testing, performed by an independent contractor under the direction of David Elliot from ED, identifies any vulnerabilities or weaknesses in existing security measures.

The server can be accessed remotely by a limited group of Raytheon developers, but only on one workstation that is outside of the Raytheon network. There are two firewalls within the VDC Virtual Private Network (VPN), which makes access to the server very difficult. Access by Raytheon personnel to remotely administer the server requires the user of a Microsoft NT User ID and a password.

9.4 NSLDS II Security

The proposed high level approach for NSLDS II application security is based on the web function a user is planning to perform. However, to transpose the existing security structure from NSLDS to NSLDS II the existing RACF file will need to be imported into a table in DB2.

This is not a detailed design of the security for the NSLDS II system. It is only a high level approach that will not cover all aspects of security (hardware, network, software, facility, and personnel).

The main security layers in this section are very similar to those in the existing NSLDS system:

- **User Identification and Authentication**—The legacy system's RACF flat file, ported to a DB2 table, will be used to verify the user's User ID and password.
- **Web Application Access Security**— Database tables are used to verify that the user is authorized to run the application from the Web, and to establish the Web pages that the user is authorized to visit.
- **Data Transmission**— Data is encrypted using SSL and server certificates.
- **Report Security**—The existing DB2 authentication table will be replicated periodically to the MicroStrategy metadata DB to verify report access.
- **EAI Security**—An EAI administrator account will be created to allow the admin to log into the EAI server, via Telnet, to make any necessary changes
- **Informatica Security**—A native connection and proprietary application level security will be used to monitor and restrict development access from the client machine to the server.

9.4.1 RACF Migration to NSLDS II:

To migrate the users and passwords from the legacy system, the RACF SMF Data Unload Utility tool will be used. This utility is available for RACF version 2.10 on the legacy system. This tool reads from the NSLDS RACF sequential flat file and migrates data from the legacy system over a DB2 connection to a relational database table on the NSLDS II system (e.g. NSLDS_USER_AUTH). The SMF Data Unload Utility tool provides the following options for the RACF sequential flat file:

- Direct viewing of the flat file.
- Input for installation-written programs.
- Manipulated with sort/merge utilities.
- Loaded into a relational database manager (for example DB2).
- Downloaded to a workstation for use with various workstation programs.

The example below displays sample JCL to invoke the RACF SMF Data Unload Utility. Refer to the RACF Macros and Interfaces for more information.

```
*/
//USER01 JOB Job card
//UNLOAD EXEC PGM=IFASMFDP
//DUMPIN DD DISP=SYS1.MANA
//DUMPOUT DD DUMMY
//OUTDD DD DISP=OLD,DSN=USER01.SMF.IRRADU00
//ADUPRINT DD SYSOUT=*
//SYSRINT DD SYSOUT=*
//SYSIN DD *
// USER2(IRRADU00) USER3(IRRADU86)
// DATE(94210)
// START(0800)
// END(1700)
// SIS(SYS1)
/*
```

Figure E. JCL for RACF SMF Data Unload Utility

Please refer to Section 2.8 of the "OS/390 Security Server - Audit Tool and Report Application" white paper for an example of how to define DB2 tables and to load RACF SMF Data Unload Utility data into these tables. There is also a member with some samples to do SQL queries to the SMF data tables.

9.4.2 User Identification and Authentication

9.4.2.1 User Logon

The user logs into the NSLDS II website by providing a user ID and password on the logon.jsp page. The User ID and password are passed via an https request encrypted at 128-bits using SSL from the client's browser through the firewall to the IHS server. The IHS server is a web server collecting client requests and mapping them to the JSP files housed on the WebSphere Application Server. The steps below are illustrated in Figure F.

1) As the user clicks submit, the logon.jsp file is directed to the ActionServlet.
2) The ActionServlet references the struts-config.xml file.
   a) The struts-config.xml file maps the logon.jsp page to a LogonForm Java Bean.
3) The LogonForm acts as both a temporary holding place and error checker for web form data. The User ID and Password are passed via the Post method from the logon.jsp page to the LogonForm, via the ActionServlet.
   a) The LogonForm performs minor error checking (e.g. valid length, characters) and returns the results to the ActionServlet.
4) If the User ID and Password are in the appropriate format, the ActionServlet passes the parameters to a LogonAction.
5) Within the LogonAction, the LogonWorker object is instantiated.
6) The LogonWorker object calls the Persistance framework.
7) The framework builds and runs a SQL query against the EDW for valid user information. See Section 1.1.3 User Authentication. The query results are returned to the LogonWorker and checked against the user's User ID and password for final validation.

If the user was successfully authenticated, the LogonAction returns an ActionForward, containing the next JSP page to be displayed, back to the ActionServlet.

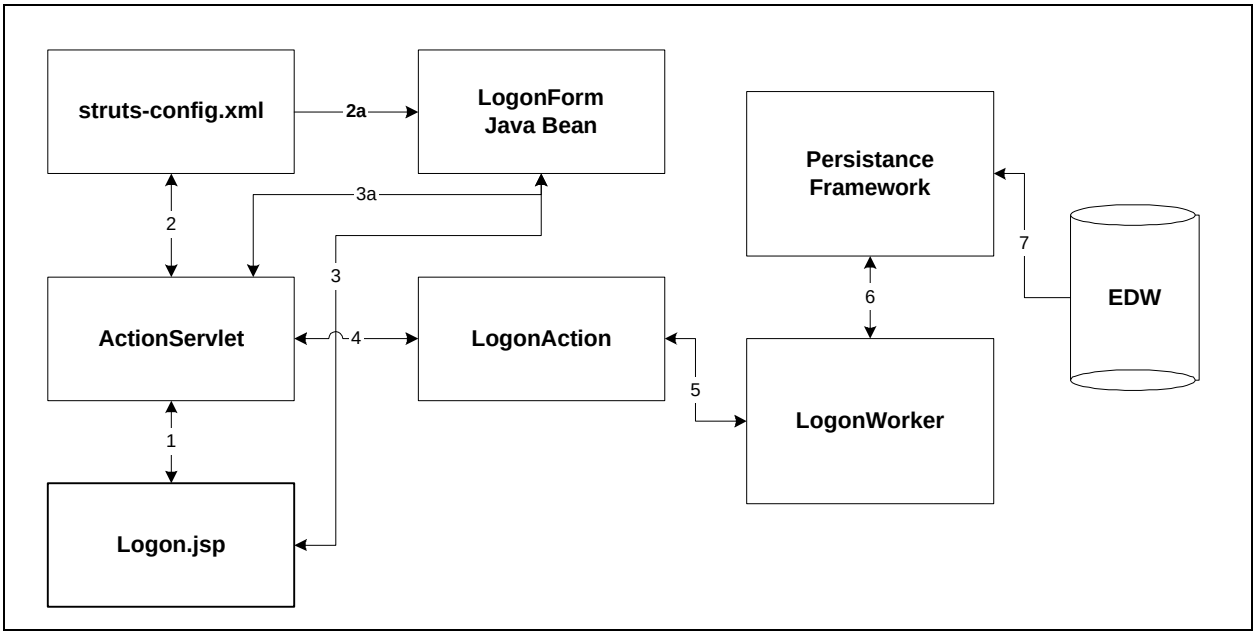


Figure F. User Logon Process

9.4.2.2 User Authentication

The User Logon process described above can occur if a connection has been established between DB2 and the WAS server. The WAS server allows for user creation and administration of the data source name via either a graphical user interface or an XML configuration file.

Specifically, the Websphere Administrative Console or XML Config file allow for the administration of a DataSource on the WAS server. The GUI interface allows the Websphere Admin to create a Data Source name (the logical Java Naming and Directory Interface name - JNDI), specify the name of the underlying database (DBMS name) and select the database driver to be utilized. It also allows for advanced parameters to be defined for connection pool size and timeouts. See section 10.3 “Administration of Data Sources” in the Websphere v3.5 Handbook.

Within the logon.jsp file a LogonAction object extends an NSLDSAction object to interact with a LogonWorker object to access the persistence layer. The persistence layer is used to obtain the requested information from the database. This code, which is distributed among these Logon objects and the Persistance Layer, creates a connection to DB2 using the DriverManager interface (JDBC 1.0), and passes a generic administrator userID and password to DB2. The SQL statement then selects the userID and password from the imported RACF table (e.g. NSLDS_USER_AUTH), and decrypts the 128-bit encrypted password column from within the table using a unique encryption password (e.g. “test123”).

Column level encryption encrypts all values in a given column using the same common password. The encryption password is set when performing a SQL insert and using the encrypt() function. This SQL statement, which simply adds a password to the DB, would most likely reside within the procedure code for the Change Password support screen. (e.g. SET encryption password = 'Test123'; INSERT INTO NSLDS_USER_AUTH (password) values(encrypt('Mypassword1'));) The user's password can later be selected and decrypted for verification so long as the encryption password is specified. The java procedure will verify that the selected information from the database matches the user's userID and password. The following code is an example of this process:

```
try {
Connection conn = DriverManager.getConnection(dbDriver,userid,password);
Statement stmt = conn.createStatement();

SQL = ("CREATE VIEW clear_auth (password) as SELECT userID, decrypt_char(password) FROM
NSLDS_USER_AUTH");
SQL = (" & Set encryption password = 'test123'");
SQL = (" & SELECT userID, password FROM clear_auth");

ResultSet rs = stmt.executeQuery(SQL)
while (rs.next()) {
String UserID = rs.getString("userID");
String Password = rs.getString("password");
}

' IF statements used here to verify that user information selected from DB matches the
' UserID and Password entered by the User.

}
catch (Exception theException) {
/ / Handle Exception
}
```

Figure G. Database Connection and Password Decryption Code

9.4.3 Web Application Access Security

User access to specific tabs, pages, and data fields (elements) will be restricted through the use of Location Groups, Function Groups and Web Groups. Location Groups define the specific location that a user is logging in from (e.g. a University's campus) and are the least common denominator for a user besides User ID. Function Groups define what functionality a user can perform (e.g. support staff functionality). Web Groups map web pages to specific function groups (e.g. access to the Support tab). In addition individual web links on a page are mapped to web groups. The following diagram lists the system security tables and their fields.

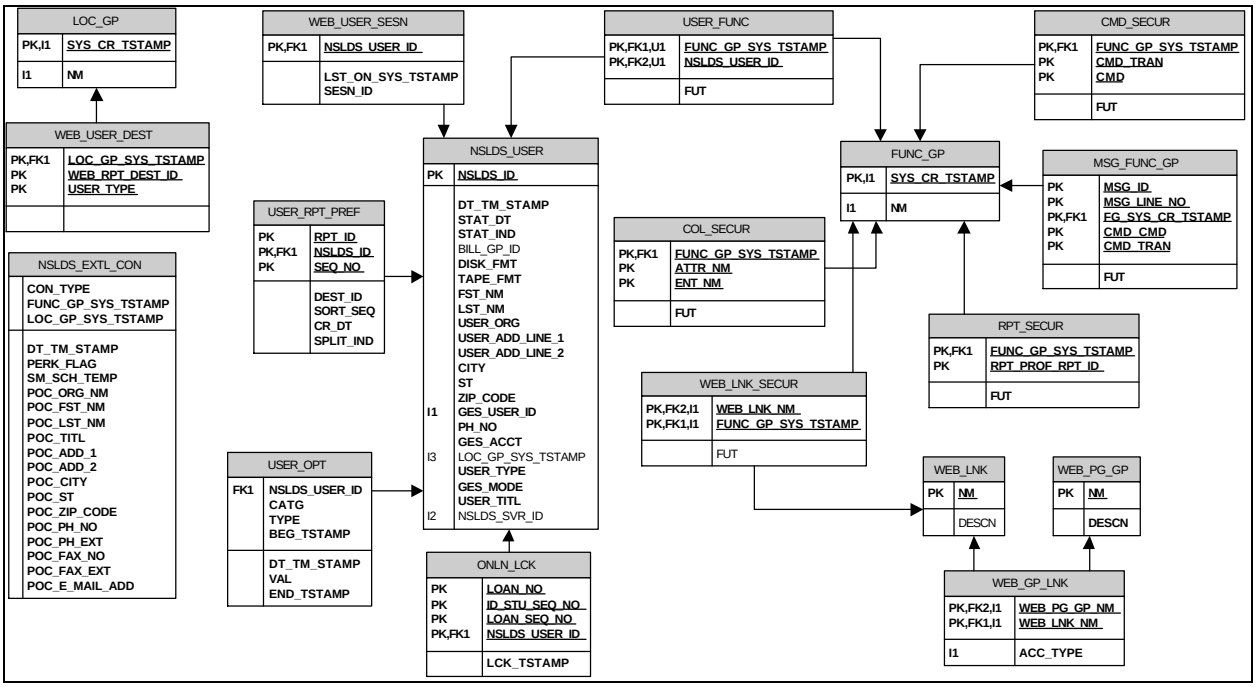


Figure H. NSLDS II Web Application Security Tables

9.4.4 NSLDS II Report Security

All user IDs and passwords will be stored in the NSLDS II EDW. This data will be imported daily from the NSLDS II EDW to the MicroStrategy Intelligence Server using the MicroStrategy Administrator – Command Manager. The Intelligence Server in turn pushes the information to the metadata layer in the MicroStrategy MetaData DB using a script that will create new users and update existing users on the MetaData DB. The script can be triggered when updates are made to the authentication table. An example of this script can be found in the Appendix. This process allows for Single Sign On functionality across the web application.

9.4.4.1 MicroStrategy and Application Integration Approach

The following diagram depicts the way Single Sign On will occur between MicroStrategy and the Web Application. The steps are outlined in more detail below the diagram.

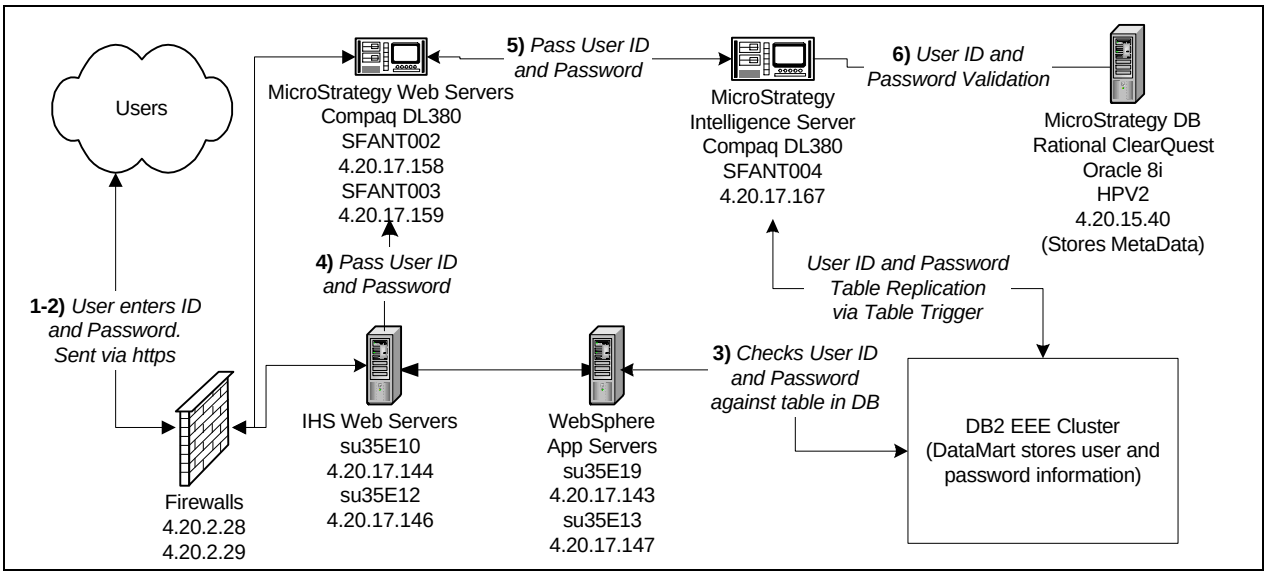


Figure I. NSLDS II Reporting Security Architecture

1. A User will enter their user ID and password on the Logon.jsp page as defined in the previous section.
2. The user ID and password will be passed via an https request over SSL to the IHS web Server.
3. The WebSphere Application Server has an established connection with the EDW that allows the web application to select the userID and password from the EDW, decrypt the password and authenticate against the parameters entered.
4. When a User clicks on the Reports Tab to create or access a report, the user ID and password will be passed in a form via a JavaScript function to the desktop.asp file on the MicroStrategy IIS Web Server. The Report Tab's link can reference the Web Server's internal NAT IP address, which will prevent the information from traversing the firewall. Thus, encrypting the user information is not necessary. However by making an https request in the link's href, SSL can be utilized for added security within the firewall if so desired.
5. The user ID and password are sent from the ASP file on the MicroStrategy Web Server to the MicroStrategy Intelligence Server.
6. The MicroStrategy Intelligence Server takes the user ID and password and checks it against the MicroStrategy metadata DB stored in Oracle 8i. The user ID and password tables have been previously replicated from the EDW to the MicroStrategy Metadata DB in the process described above.
7. If the user ID and password exist, the MicroStrategy Home page is displayed in a separate browser window. The user is now accessing MicroStrategy ASP files directly from the MicroStrategy IIS Web Server. The requests are encrypted via SSL.

8. The user is allowed to either create a new report or view a previously run report based on his/her access rights.

9.4.4.2 Authentication and Authorization

The user will select the Reports tab in the NSLDS II website and will be routed to the MicroStrategy desktop.asp page. From the Desktop page the user can click on "Shared reports" to view a list of possible reports that he/she may run. The user is routed to the Folders.asp page where only those reports to which the user has been granted access will be listed on the page. The user can also click on the "History List" link to view the results of the reports that were previously run. The user is taken to the Inbox.asp file when selecting History List. Function Group assignments define which individual reports a user may generate and/or access. (See Appendix A Sections 3.5 or Reference: Report Content Descriptions with Purpose v1.0 APW.xls for a list of Function Groups)

The MicroStrategy Shared Reports page (i.e. Folders.asp) is broken down into 2 frames. The first (top) frame is populated with existing MicroStrategy reports. The user can select a report, enter the necessary information and run the report with submitted parameters.

The second (bottom) frame is populated with a list of reports that exceed MicroStrategy's technical capability (large format reports). These reports are referred to as "exception reports."

Should the user click on an "exception report" link within the bottom frame of the Home Page, the page will load an ASP page in a new browser window. The User can then enter querying parameters that are specific to each exception report. This page saves the parameters to a table on the EDW. Table population triggers an Informatica stored procedure that queries the database for the required information and in turn creates and stores the information in a temporary table. The stored procedure will then run a Java executable from the db2 command that will pull the information from the temporary table and generate a formatted report file. The report files are stored locally on the DB2 server. To download the report files, the user will need to navigate to the History List page.

The History List page allows a user to view the reports run on the Shared Reports page or to download an exception report file. The History List page (i.e. Inbox.asp) is broken into two frames. The top frame allows the user to view a previously run MicroStrategy report simply by clicking on the report name. The bottom frame displays a dropdown box that allows the user to select their "exception report" from the list populated with all the available exception reports to which the user has been granted access. The user can then click the "Save As" button, and save the file locally. The user is not allowed to see where the file was stored on the DB2 server nor the server's IP address.

9.5 NSLDS II EAI Security

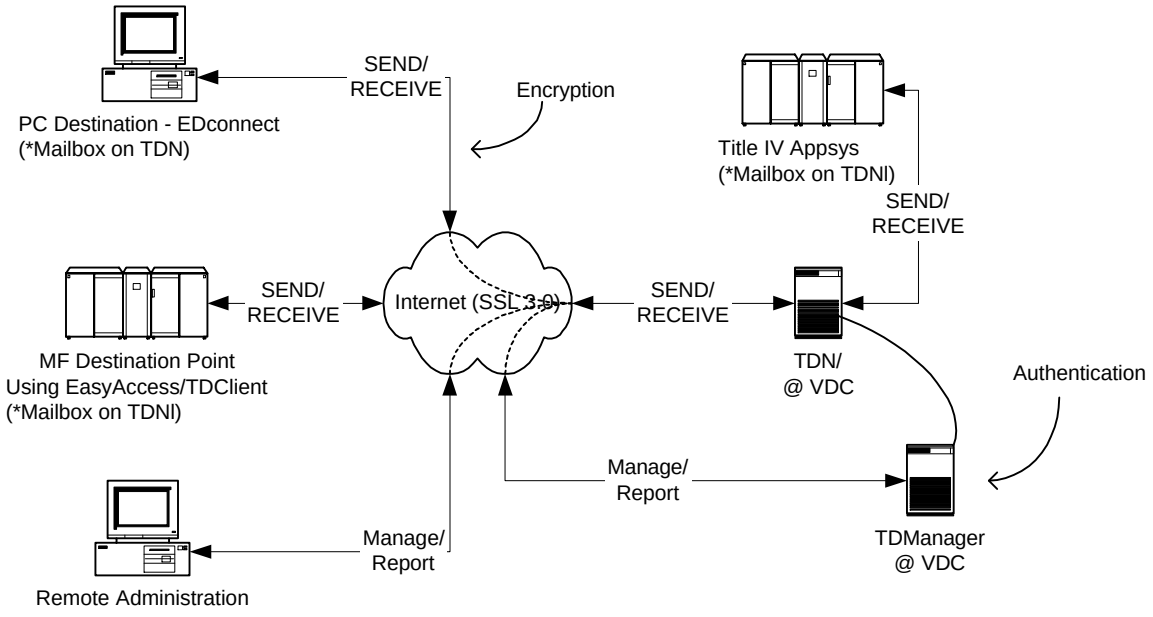
9.5.1.1 EAI Security

Since the EAI bus and its appropriate adapters are installed, and collect information from mailboxes and other trading partners within the VDC's firewall, the EAI component does not have any secure data transmission functionality beyond signing into the EAI bus for EAI administrative changes. The EAI administrator logs on via Telnet to perform any necessary changes. He/she must have an administrative account on the EAI bus to logon.

The files processed by EAI are first transferred to the VDC via the Student Aid Internet Gateway's (SAIG) existing security procedures. SAIG offers Title IV-eligible post-secondary institutions, third-party servicers, state agencies, lenders and guarantors a secure, Internet-based method of exchanging Title IV data with the FSA Application Systems.

SAIG uses a system of mailboxes and a secure FTP client (i.e. "EasyAccess") to transfer files from the user (e.g. data provider) to a mailbox for the system within the VDC (e.g. the "NSLDS" mailbox). In turn the EAI bus and its appropriate adapter retrieves the file from the mailbox and processes the file.

The diagram below has been taken from "SAIG: Host Communication Guide for Mainframe and Midrange Users Version 1.4", it illustrates the SAIG secure file transfer system.



* SAIG destination points and SAIG Application Systems using EasyAccess/TDClient software component. EasyAccess/TDClient incorporates the security features of TDManager and provides data security functions (authentication, encryption) to SEND/RECEIVE data to TDN. PC destination points continue to use EDconnect software (EDconnect will integrate EasyAccess/TDClient API).

Figure J. SAIG Security Diagram

9.5.2 NSLDS II Informatica Security

The Informatica administrator uses client software installed on his/her machine to connect to the Informatica server within the VDC. Informatica uses a native connection and application level security to monitor and restrict development access from the client machine to the server. This application level security is outlined in Appendix A Section 3.4 Informatica Application Level Security.

Given that the actual data load occurs within the firewall from the EAI bus to the Informatica Server to the EDW, as well as from the EDW to the Informatica server back to the Data Mart, data encryption is not necessary for data transfer over the TCP/IP connection.

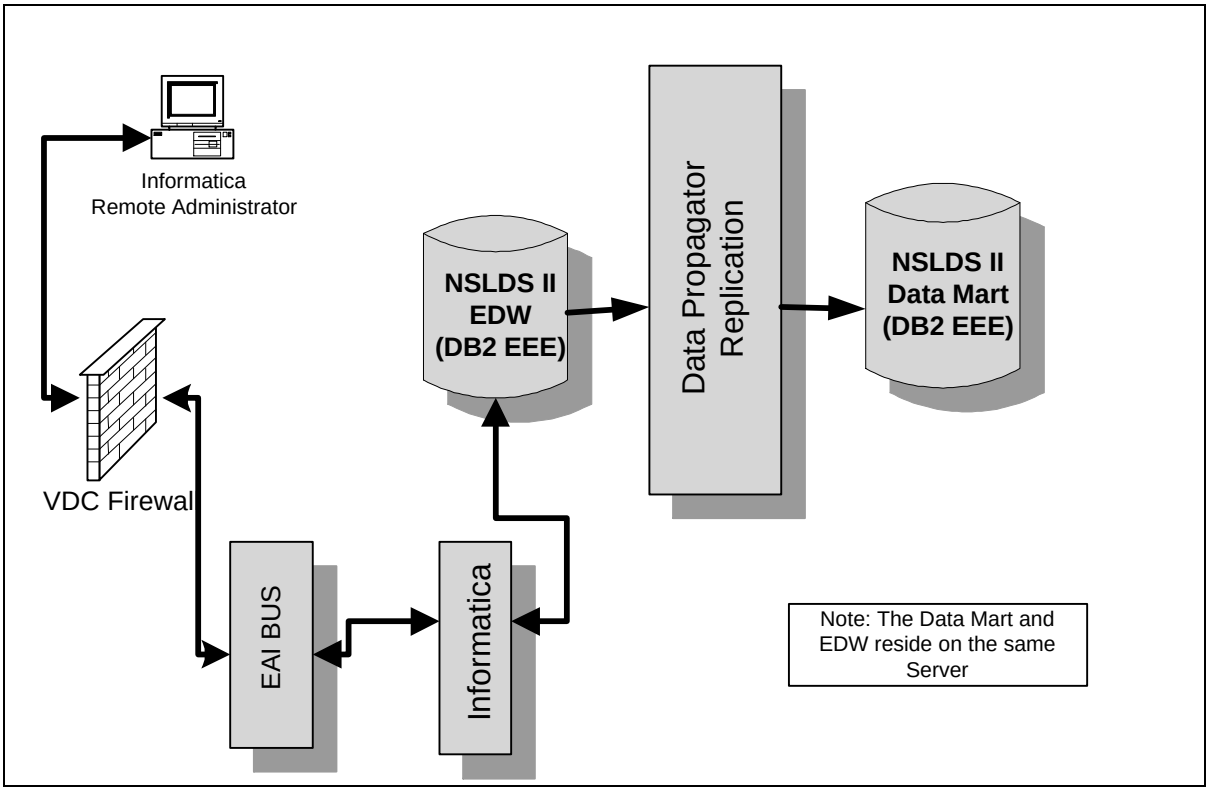


Figure K. Informatica Security Diagram

10 NSLDS II Configuration Information

The following sections detail the configuration of the NSLDS II operating system and database environment for the development, Test1, Test2, and Production architectures. Since the development environment was the only environment fully built out during the detailed design phase of NSLDS II, the development configuration section is the most detailed. The Test1, Test2, and Production designs are differentials between the detail of the development environment and their configuration. As additional information is gathered during the build phase of NSLDS II, the designs for Test1, Test2, and Production will be updated accordingly.

10.1 Development Environment

Figure E details the current design of the NSLDS II development environment. Note that for this detailed design document, the configuration information only applies to the "NSLDS II Data Architecture" server and array located on the right hand side of the diagram.

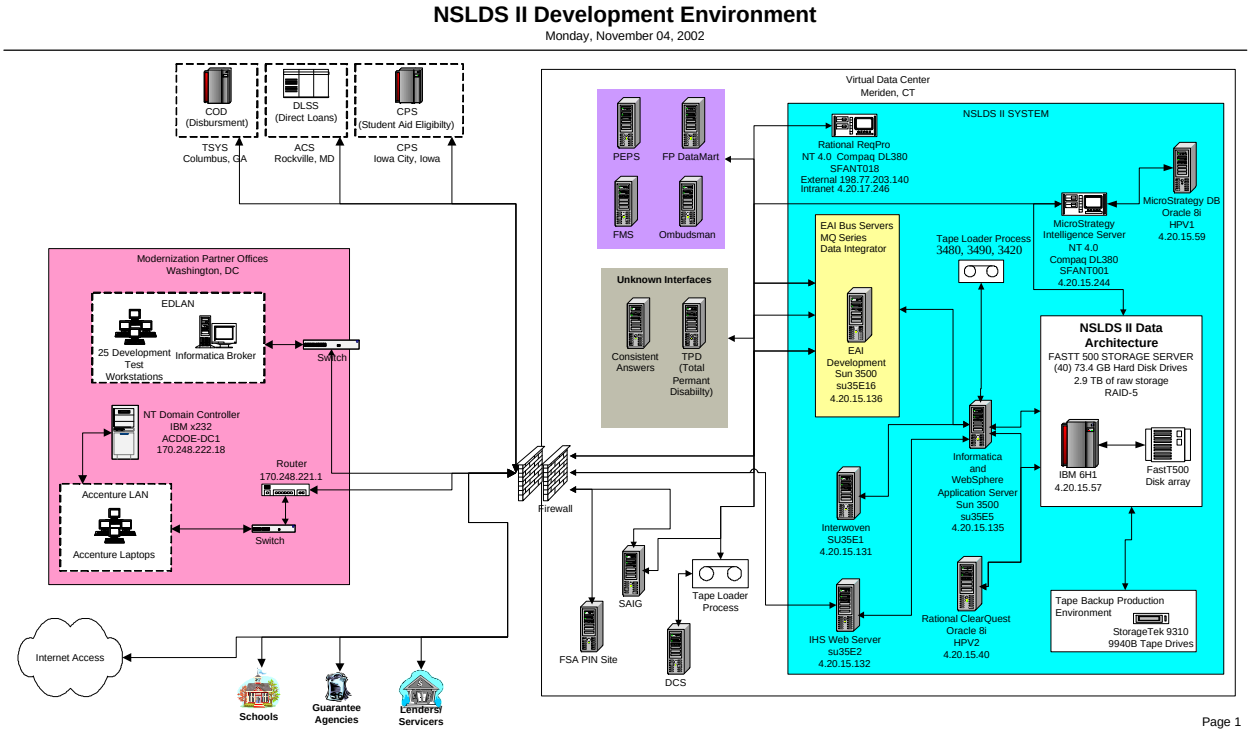


Figure L. NSLDS II Development Hardware Architecture

The database server is an IBM pSeries 660 model 6H1 with four 750MHz processors, 4 GB of memory, two Internal 36.4GB disks, and a FastT500 disk array with forty 73.4 GB hard drives (2.9 TB). The operating system is AIX version 5.1 with DB2 version 7.2 EEE (Please reference the NSLDS II Reengineering Operations Architecture Plan for further configuration information).

10.1.1 AIX Operating System Environment

This section details changes required to the AIX environment to support a DB2 UDB EEE installation on the development database server (IBMP1). These changes are required to either support the basic installation on the system or provide system stability.

10.1.1.1 Update to /etc/services File

To simulate production as closely as possible, the DB2 UDB EEE environment is configured to run with up to seven logical partitions with the following change made to the /etc/services file:

```
DB2_db2inst1        60000/tcp
DB2_db2inst1_END    60006/tcp
```

The format must always be **DB2_<instance name>** and **DB2_<instance name>_END**
The port range must not be used by any other applications.

10.1.1.2 User Limit for User db2inst1

The user limit for user db2inst1 was changed so that there would be no possibility of collision of the stack and data areas in segment #2 in a 32-bit AIX operating environment. In a 32-bit AIX environment, both the data and stack areas come out of the same 256-megabyte segment (segment number two). The stack and data start out at opposite ends of the segment and move towards each other. It is imperative that the values set for stack and data not be able to cause overwrites in the segment. It is also important to have the data segment as large as possible without overwriting the stack to allow large DB2 UDB SORTHEAP values to be allocated (as they come out of this data area). For the development IBMP1 system, the values were changed to:

```
data    229000 1K pages
stack   32768  1K pages
```

These values are what should be seen if issuing “ulimit –a” from the AIX command line while logged in as the userid **db2inst1** (If viewing the user limits from within the AIX SMIT utility, the SMIT indicates the values in 512-byte increments, while ulimit indicates the values in 1K increments).

10.1.1.3 MAXUPROC for AIX Operating Environment

The maximum number of processes for a user was increased from the initial 1000 to a value of 2000, which is more appropriate for an initial DB2 UDB EEE environment.

10.1.1.4 RAID Array Information

The AIX CSC team has configured the development FastT500 array in a 9+1 configuration. This translates to 9 useable disks per LUN, which is a consideration when determining the pre-fetching information for the table spaces.

The stripe size for this configuration, per the AIX CSC team, is 16K. This value should be considered when determining the table space extent size.

10.1.1.5 AIX File Systems for use by DB2 UDB EEE

From discussions with the CSC AIX team, the following decisions for the development DB2 server (IBMP1) were made:

- The existing available disk space of approximately 2-terabytes will be reserved for the DB2 UDB EEE development database (P1DWDV01) although only the first 1-terrabyte of space will be used for the initial configuration.
- The DB2 UDB will have forty file systems for use.
- There will be 10 file systems per each of the 4 LUNs.
- Naming convention for the file systems will be: /u01, u02, … /u39, /u40
- Each file system was created with the large-file option enabled.
- Each file system was given ownership of user: db2inst1 group: db2iadm1

The four available LUNs are represented as hdisk3, hdisk4, hdisk5 and hdisk6. Each has 10 disks in a 9+1 configuration (9 data plus a spare). The mapping of the file systems to LUNs is the following:

hdisk3	hdisk4	hdisk5	hdisk6
/u01	/u02	/u03	/u04
/u05	/u06	/u07	/u08
/u09	/u10	/u11	/u12
/u13	/u14	/u15	/u16
/u17	/u18	/u19	/u20
/u21	/u22	/u23	/u24
/u25	/u26	/u27	/u28
/u29	/u30	/u31	/u32
/u33	/u34	/u35	/u36
/u37	/u38	/u39	/u40

For the process of mapping the file systems to a partition, DB2 version 7.2 EEE has an Everything Across Everything approach. Under this configuration, the table spaces are spread across more than one LUN with a rule-of-thumb as six to ten disks per CPU. Since the development environment has 40 available disks and 4 CPUs, it was decided to use all LUNs for all data partitions to spread the load. In the other NSLDS II environments that have more disks, the LUNs may be separated such that each is allocated to only one DB2 UDB partition. Additionally, the catalog partition data and the database logs could be separated out onto

multiple LUNs or other disks. This will give the benefit of totally separating access to data for one partition from another partition.

10.1.1.6 Paging space

Verified that the existing paging space is equal to the existing memory of 4-gigabytes. If the memory is increased on the development server, the paging space should be increased accordingly to maintain the relationship of paging space size equal to real memory.

10.1.1.7 User Name Length Restrictions

The AIX *userid* has a limitation of 8 characters for its total length. This must be taken into account when creating common *userid*s across various operating platforms in the environment.

10.1.1.8 Auto-starting the DB2 UDB Instances on AIX Startup

Depending on the requirements of the development team, DB2 UDB EEE can be set to auto-start on boot up. To do this, the following must be done:

- The file rc.db2 must be included in the /etc directory on the AIX box and must not be modified from the delivered version from IBM.
- The following entry must be placed in the /etc/inittab file on the AIX box:

  db:2:once:/etc/rc.db2 > /dev/console 2>&1 # Start DB2 services

- For each DB2 UDB instance you wish auto-started at AIX boot time you must set the AUTOSTART registry setting for that instance (see DB2 UDB Registry Variables below).

Note that the DB2 Administration Server instance (db2as) is always auto-started and no additional registry settings are required.

10.1.1.9 Configuring the DB2 Universal Database Enterprise-Extended Edition

10.1.1.9.1 Update to DB2 UDB Version 7.2 Fix Pack Seven

The environment should be running the latest fix pack (at time of writing) fix pack seven. Both the db2inst1 and db2as instances were updated from fix pack three that was initially installed on the development server to the recommended level. Note that it is recommended that all clients accessing the DB2 UDB server also be upgraded to the same level of DB2 UDB code (at this time, fix pack seven).

10.1.1.9.2 DB2 UDB Security

It is recommended that the default security level of *Authentication = SERVER* be maintained. This will require that all users who wish to attach to the DB2 UDB instance db2inst1, or connect to the database P1DWDV01, be authenticated at the DB2 UDB EEE AIX server. One caveat is that in order to do this all *userids* must be defined in the AIX operating system on the DB2 UDB EEE server as DB2 UDB EEE uses the operating system for its userid and password authentication. Further privileges can be determined on a per user basis (such as which tables a user can see, etc.) by using the SQL GRANT and REVOKE statements to indicate which

privileges against database objects a particular userid should have.

In addition, the system has the flexibility to grant privileges to user groups via first setting up users in AIX groups and then using DB2 to set the GRANT/REVOKE database privileges to this AIX group instead of setting security for individual users.

10.1.1.9.3 Mapping of Database Tables to Database table spaces

All tables will be assigned to table spaces in a manner consistent with the EDW Data Model. For instance, all tables associated with the **Students** model element will be stored in either a single partition table space called **STUDENTS_SINGLE** or a multiple partition table space called **STUDENTS.** The determination of which table space a table will be placed in (single partition or not) will be based on the number of expected rows in that table. For the development environment, the rule-of-thumb will be if the table row count is 100,000 rows or less, then it will not be partitioned.

Again DB2 UDB EEE has the flexibly to move a single partition table to a multiple partition table if a table grows significantly. It may also be useful to replicate the smaller table on each partition to try to achieve collocated joins if the partitioning keys allow for this.

10.1.1.9.4 Partitioning Information

Initial partitioning key definitions for the development environment have been set to be the same as the unique index on the table to expedite the creation of the development environment. This is by no means the optimal definition and will be modified moving forward from the development environment into the test1, test2 and production environments as time is made available for analysis of join columns used between tables, etc.

For the purposes of the development environment, it was determined that a row count of 100,000 rows would be the cutoff to determine if a table should be partitioned or not. Any tables with row counts less than or equal to 100,000 would be placed in the single partition nodegroup. All others would be partitioned.

In an attempt to emulate a production environment, the development environment was created with four logical DB2 UDB EEE partitions (A single catalog partition (zero) and three data partitions (one, two and three)).

10.1.1.9.5 Indexing and Referential Constraints

The indexes and the referential constraints defined in the NSLDS II database environment are taken from the existing DB2 for MVS environment and converted to DB2 UDB for AIX SQL statements. Where not already specified by DB2 for MVS, the primary key definitions for the tables are created to use the existing UNIQUE index structure and hence will use the UNIQUE index for the primary key constraint. This is done by specifying the creation of the UNIQUE index first and then altering the table definition to indicate a PRIMARY KEY should be added

with the same columns as the UNIQUE index. DB2 UDB will note that the columns in the PRIMARY KEY are identical to those in an existing unique index and will not create another UNIQUE index to support the PRIMARY KEY constraint. Rather it will use the existing UNIQUE index to support the constraint defined.

10.1.1.9.6 Instance (DBM) Registry Settings

DB2 UDB registry variables are settings at the DB2 UDB instance level and system level, which affect all databases within the instance (if defined at the instance level) or system (if defined at the system level). All databases under the **db2inst1** instance will be affected by these settings. That is to say that if the Data Mart is created under this same instance it will inherit these same settings. If the Data Mart is created under a different instance, then the instance level registry variables will need to be applied to that instance. It is recommend setting the following DB2 UDB registry settings for the development environment.

Registry Variable Settings	Description
db2iauto -on db2inst1	Set to auto start "db2inst1" on AIX startup.
db2set DB2TCPCONNMGRS=1	Set to only have 1 TCP/IP connection manager.
db2set DB2MEMDISCLAIM=ON	Set to reduce amount of memory held by idle DB2 agents.
db2set DB2MEMMAXFREE=4000000	This setting is for the amount of memory an idle agent should allocate.
db2set DB2_PARALLEL_IO=*	Set to enable parallelism when accessing containers on more than 1 disk.
db2set DB2_STRIPED_CONTAINERS=ON	Set to use a full extent for a container tag in a RAID array environment.
db2set DB2_HASH_JOIN=YES	Enables HASH joins in access plan analysis.
db2set DB2_UPDATE_PART_KEY=YES	Set to allow partitioning keys to be updated.
db2set DB2_FORCE_FCM_BP=YES	Set to use a shared memory segment for inter-partition communication rather than TCP/IP sockets.

Figure M. NSLDS II Development DB2 UDB EEE Registry Settings

10.1.1.9.7 Instance (DBM) Configuration Parameters

The database manager level (DBM or instance) configuration parameters affect the running of the DB2 UDB instance and all databases defined within that instance. For the development environment, the following DBM (instance) configuration parameters were updated:

DBM Parameter Setting	Description
numdb 4	Set to allow up to 4 databases in the db2inst1 instance
dft_mon_lock on	Turns on a default set of monitor switches to allow easier monitoring
dft_mon_bufpool on	Turns on a default set of monitor switches to allow easier monitoring
dft_mon_sort on	Turns on a default set of monitor switches to allow easier monitoring
dft_mon_stmt on	Turns on a default set of monitor switches to allow easier monitoring
dft_mon_uow on	Turns on a default set of monitor switches to allow easier monitoring
mon_heap_sz 256	Set to increase the monitor heap size for snapshot monitoring
maxagents 800	Set for the number of DB2 agents that can execute per partition
sheapthres 200000	Set to increases the soft limit on the max amount of sort heap that can be allocated at one time. In an environment with INTRA_PARALLEL OFF, this is a soft limit; otherwise it is a hard limit .This value will in effect limit the number of sorts that can occur concurrently. For instance, if the SORTHEAP is 20000 4K pages and the SHEAPTHRES (Sort Heap Threshold) is 200000 4K pages, it implies that the database can have 10 concurrent sorts per partition.

Figure N. Instance (DBM) Registry Settings

10.1.1.9.8 Default Database Outline for Development Database

Please see *Appendix SQL for Development* for all scripts and files used to create the development database on IBMP1 and for a listing of DB2 SQL files used to create the various tables in the database.

10.1.1.9.8.1 EDW Database Configuration

The base outline for the development EDW database was created with the following parameters:

- When the database is created, the first partition is set to zero by default. For the development environment configuration, the zero partition is the catalog partition and the default coordinating partition of the database.
  - For a larger database configuration, the coordinating partition may be separated out into its own partition to increase performance. In this scenario, the coordinating partition would not contain any data but would only be used for DB2 UDB coordinating agents and would have memory and CPU available to it.
- The system catalog table space is created on file system /u01 and has four containers created to minimize the possibility of AIX i-node latching issues. This table space is created as System Managed Space (SMS), which is the recommended for the system catalog table space. The following are the configuration changes:

  - The P1DWDV01 database is created such that no other user data is placed on the catalog partition. This is very important for recoverability in the event the database is abnormally terminated and will speed up the recoverability of the catalog partition. This will allow faster recoverability of the remaining partitions.

- The user and temporary table spaces, while initially created on /u01, are only created as placeholders. The user table space USERSPACE1 will be dropped later in the overall process and the temporary table space TEMPSPACE1 will be replaced with another. The following are the configuration changes:
  - The USERSPACE1 user data table space was dropped to restrict users from being able to create tables in a default table space.
  - For TEMPSPACE1, a new temporary table space is created and mapped across all LUNs.
- The "db2empfa" command is executed on all partitions in order to allow the SMS table space dynamic allocation (this is done by extent number of pages at a time rather than 1 page at a time, which is the default). This will benefit the overall performance and only affects the USER and System Catalog table spaces. It does not affect TEMPORARY table spaces as defined as SMS. It should also be noted that once set the table spaces cannot be unset.
- Log paths for the four partitions are on separate file systems: /u02, /u03, /u04, /u05.
- Database configuration parameters for all partitions are as follows:
  - MAXFILOP for an agent is set to 200
  - Number of I/O cleaners is set to a value of 1 (# CPUs/ # partitions)
  - Number of I/O servers is set to a value of 40 (number of disks)
  - Increase the UTILITY HEAP (used for backup/restore/runstats/load) to a value of 20000
  - Change the lock timeout from infinite (-1) to 120 seconds
- On all partitions, except the catalog partition, the following changes are made (the catalog partition, not holding user data, should not need to have these parameters modified as there should not be user data on it):
  - Increase lock list size to 40000
  - Increase the statement heap to 4096
  - Increase the application heap size to 256
  - Increase the statistics heap size to 6144
  - Change the average number of unique applications running against the partition to 5
  - Increase the SORTHEAP available per SQL statement sort request to 20000 4K pages
  - Change the log file size to 40000 4K pages
  - Change the number of primary log files to 30
  - Change the number of secondary log files to 30
- On the catalog partition the following parameters apply:
  - Change the log file size to 10000 4K pages
  - Change the number of primary log files to 30

  - Change the number of secondary log files to 30

- Two nodegroups are created for the development environment:
  - A single partition nodegroup-encompassing partition 1 and called *p1dwdv01_ng_single.*
  - A multiple partition nodegroup encompassing partitions 1,2,3 and called *p1dwdv01_ng_multi*
  - It was decided to create a new multiple partition nodegroup on partitions 1,2,3 to separate the user data from the catalog partition. If the default IBMDEFAULTGROUP nodegroup had been used for partitioned user information, data would have been placed on the catalog partition by default. While it is possible to remove a partition from the IBMDEFAULTGROUP, it is a better method to create a new nodegroup and use that.
- A new temporary table space was created for all partitions using all of the LUNs. This table space is a replacement for the initial TEMPSPACE1 table space, which is then dropped, and the new one renamed to be TEMPSPACE1.

- As generic users will not be able to create their own tables, the default data table space USERSPACE1 is dropped.

- Two user temporary table spaces are created; one in the single partition nodegroup the other in the multiple partition nodegroup. These are created as SMS table spaces, which is the recommended format for temporary table spaces. These table spaces will allow users or applications to DECLARE their own GLOBAL TEMPORARY TABLES for use within their connection session. These tables do not log nor do they appear in the DB2 UDB system catalog tables. They exist only for the duration of the connection session**.**

- The buffer pool sizing will be done on the database when development starts and data is loaded into the system. The 1.5-gigabyte limit per partition of database-shared memory for DB2 UDB V7.2 should be taken into account for any configurations.

- The table spaces for all table data are created based on the information in *Appendix B EDW and Data Mart Data Models.* Tables are grouped based on the data model with the name of the table spaces based on the data model entity. There are two table spaces per each entity grouping. One is for the multiple partition nodegroup and one is for the single partition nodegroup. With this design, it is easy to correlate tables, which relate to the same model entity in either the single or multiple partition nodegroups. Additionally, tables with row counts of less than 100,000 rows will go into the single partition nodegroup and others in the multiple partition nodegroup. Non-partitioned tables in the single node nodegroup include:

Table Name	Table Space Name	Description
def_rates	def_rates_single	Default Rates entity
fdslp	fdslp_single	FDSLP entity
guar_agenc	guar_agenc_single	Guaranty Agencies entity
lenders	lenders_single	Lenders entity
misc	misc_single	Miscellaneous entity
schools	schools_single	Schools entity
students	students_single	Students entity
transf_mon	transf_mon_single	Transfer Monitor entity
loans	loans_single	Loans entity

Figure O. Non-Partitioned Table Space Names

- For the table spaces, the extent size is chosen to be a multiple of the RAID array stripe size (which is 16K). The extent size is chosen to be 64K or 16-4K pages.

- For the development database, the prefetch size is initially set to 144 - 4K pages, which is the extent size of 16-4k pages multiplied by the number of disks in a LUN that are used for a table space container. This is an **initial** setting only and it is expected that this value will be increased to allow for a higher level of prefetching once performance tuning is started.
  - The PREFETCHSIZE size optimally should be set to a multiple of the EXTENT SIZE where the multiplier is the number of useable disks that the table space containers are spread across.
  - The Prefetch size can be modified at any time by using the ALTER TABLESPACE statement to change the prefetch size for a particular table space.

- As development commences, tables that are in the single partition nodegroup may be collocated to other user data partitions, as it may be beneficial to SQL join plans.

- All tables have added the column *DT_TM_STAMP* with the parameter *NOT NULL WITH DEFAULT*. This sets the field so that anytime a new record is inserted into the table, if the DT_TM_STAMP field is not supplied, the *CURRENT TIMESTAMP* will be used to populate the field of the record being inserted. Additionally, the *DT_TM_STAMP* field can be updated using SQL at any time to be the *CURRENT TIMESTAMP* by using the *CURRENT TIMESTAMP* special register.

10.1.1.9.8.2 Data Mart Database Configuration

The base outline for the development Data Mart was created with the following parameters:

- Instance Configuration: The Data Mart database will be created in the same instance as the EDW database.

- The system resources like Sort memory. Buffer pool, etc will be separate from EDW database and controlled through database configuration parameters. The following Data Mart database configuration parameters will be changed:
  - On all partitions the following changes will be made:
    - MAXFILOP for an agent is set to 200.
    - Number of I/O cleaners is set to a value of 1 (# CPUs/ # partitions).
    - Number of I/O servers is set to a value of 40 (number of disks).
    - Increase the UTILITY HEAP (used for backup/restore/runstats/load) to a value of 20000.
    - Change the lock timeout from infinite (-1) to 120 seconds.
  - On all partitions, except the catalog partition, the following changes will be made (the catalog partition, not holding user data, should not need to have these parameters modified as there should not be user data activity on it):
    - Increase lock list size to 40000
    - Increase the statement heap to 4096
    - Increase the application heap size to 256
    - Increase the statistics heap size to 6144
    - Change the average number of unique applications running against the partition to 5
    - Increase the SORTHEAP available per SQL statement sort request to 20000 4K pages
    - On the catalog partition
    - Change the log file size to 10000 4K pages
    - Change the number of primary log files to 30
    - Change the number of secondary log files to 30
    - Change the log file size to 40000 4K pages
    - Change the number of primary log files to 30
    - Change the number of secondary log files to 30
- For the Data Mart node groups, there will be two node groups created:
  - *mart_ng_single* – This will be on a single partition (1) to create all single partition table spaces
  - *mart_ng_mutli* – This will be on all the partitions, number 1,2,3 to create all multi-partition table spaces.
- Data Mart table spaces will be designed such that each subject area will have two tablespace configurations:
  - To accommodate non-partitioning tables within subject area, the table space will be on a single partition node group "*mart_ng_single*".
  - To accommodate large partitioning tables within a subject area, the table space will be on a multiple partition node group "*mart_ng_multi*".
  - The distinction between where a table belongs on the node groups is dependent on the size of the table. If a table in the Data Mart is greater than 250 MB or if there is multiple join requirement from a query or report, partition on a multiple partition node group. A detail list of all Data Mart tables and corresponding size for the partition strategy is described in *Appendix F Appendix F Capacity Planning for NSLDS II.*

- The development Data Mart database will be created with following single partition table spaces and multi-partition table spaces:

Single Partitions	Multiple Partitions
SINGLE_GA	GA
SINGLE_LENDER	LENDER
SINGLE_SCHOOL	SCHOOL
SINGLE_STUDENT	STUDENT
SINGLE_SCCR	SCCR
SINGLE_OTHER	OTHER

Figure P. NSLDS II Data Mart Single and Multi-Partition Table Spaces

  - The OTHER and SINGLE_OTHER will hold smaller subject areas like GA_SUM, CDR, PRESCRRENING, ORG_CONTACT, MISC_LOOKUP etc.

- A new temporary table space will be created. It will be created for all partitions using part of all of the LUNs. This table space will be the replacement for the initial system TEMPSPACE1 table space, which then will be dropped, and the new one renamed to TEMPSPACE1.

- Two *user* temporary table spaces will be created; one in the single partition nodegroup the other in the multiple partition nodegroup. These will be created as SMS table spaces, which is the recommended format for temporary table spaces. These table spaces will allow users or applications to DECLARE their own GLOBAL TEMPORARY TABLES for use within their connection session. These tables do not log nor do they appear in the DB2 UDB system catalog tables. They exist only for the duration of the connection session**.**

- The Indexing Strategy for the Data Mart will be created with primary key index and foreign key indexes. Additional indexes will be created once development has started and an analysis of the data usage and performance of adhoc queries can be gauged.

10.2 Test 1 Environment

The Test1 database server is primarily used for integration and system testing, Since this environment has the same hardware configuration as the development environment (CPU, Memory, Capacity), the parameters for this environment will be very similar. Figure I details the current design of the NSLDS II Test1 environment. Note that for this detailed design document, the configuration information only applies to the “NSLDS II Data Architecture” server and array located on the right hand side of the diagram.

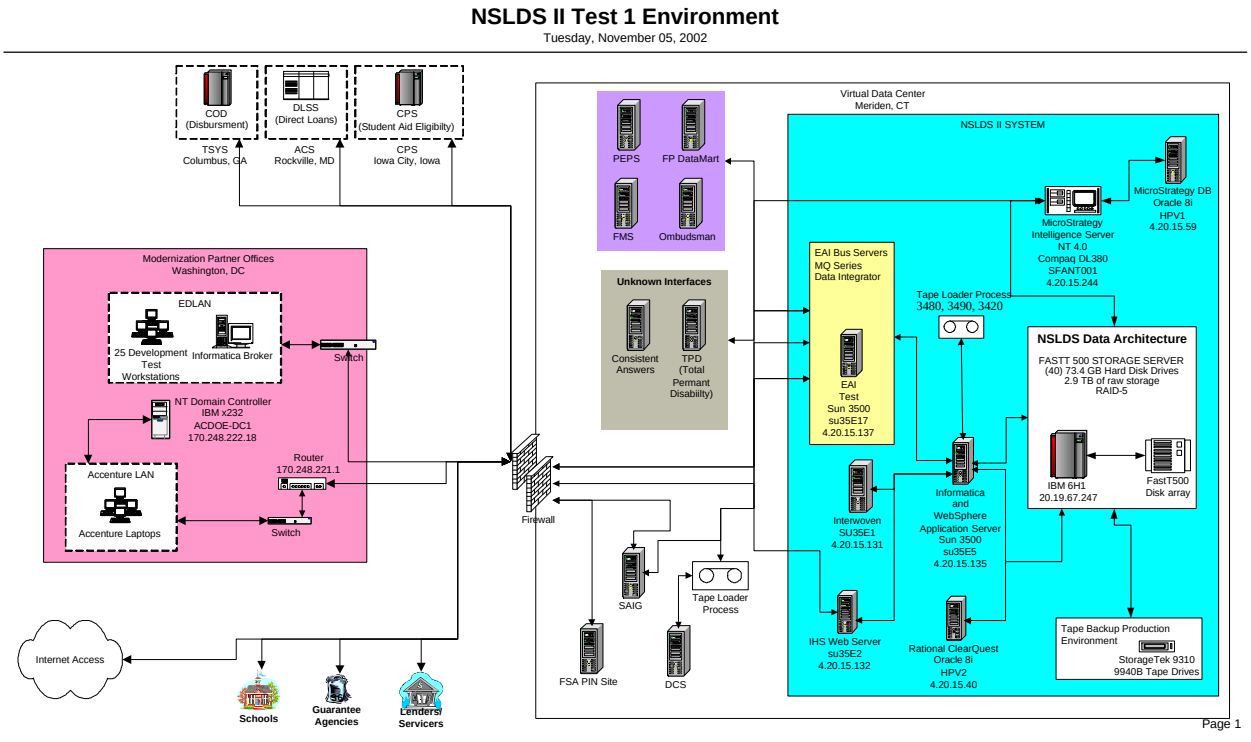


Figure Q. NSLDS II Test 1 Hardware Architecture

For this environment, the database server is an IBM pSeries 660 model 6H1 with four 750MHz processors, 4 GB of memory, two Internal 36.4GB disks, and a FastT500 disk array with forty 73.4 GB hard drives (2.9 TB). The operating system is AIX version 5.1 with DB2 version 7.2 EEE (Please reference the NSLDS II Reengineering Operations Architecture Plan for further configuration information).

Since the server has not yet been built out for this environment, the existing development configuration will be used as a starting point. No additional parameters will be defined at this time.

10.3 Test 2 Environment

The Test 2 database server will be used primarily for system testing, training, and inter-system testing (interfaces). Additionally, the Test 2 environment will be used at times for performance testing and some preliminary conversion testing. This system will be configured similar to the production environment in that IBM's High Availability Cluster Multiprocessing (HACMP) technology will be implemented. Unlike production, the Test 2 environment will consist of only two IBM pSeries 660 model 6H1 servers with four 750MHz processors, 4 GB of memory, two Internal 36.4GB disks, and a FastT500 disk array with one-hundred 73.4 GB hard drives (Since there are two arrays, this is equivalent to approximately 14 TB of storage for multiple arrays).

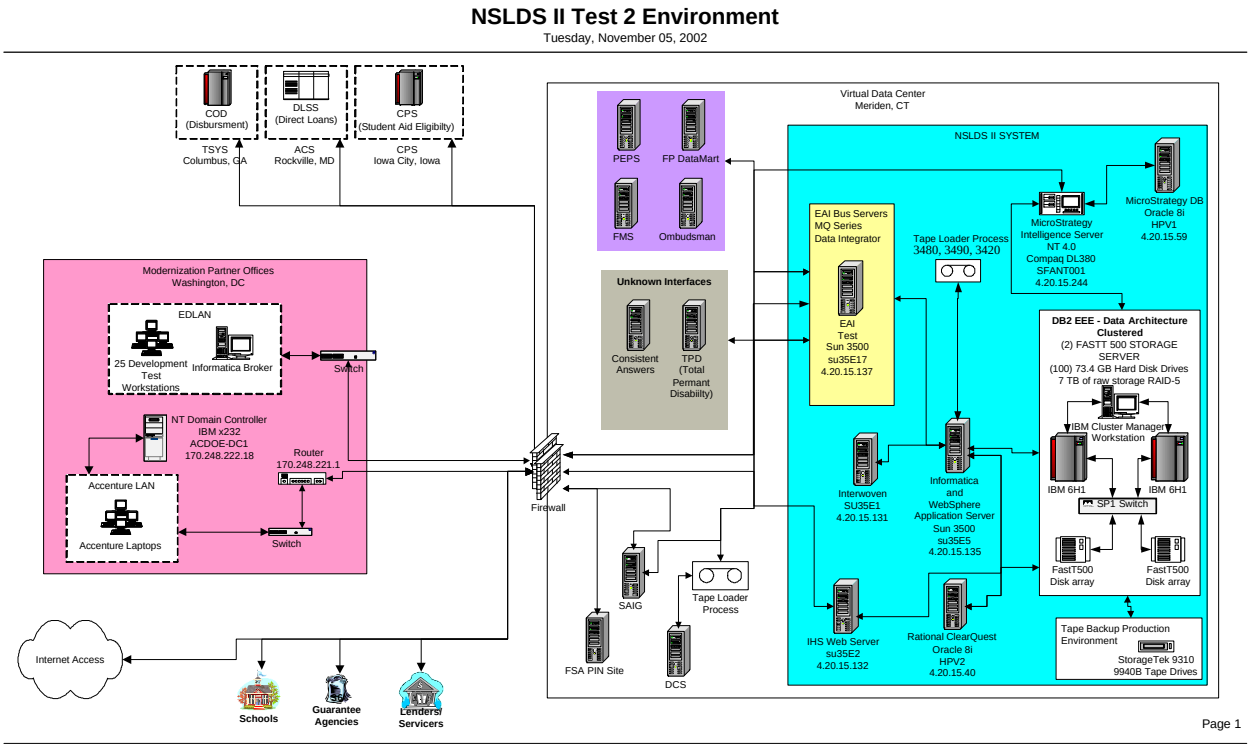


Figure R. NSLDS II Test 2 Hardware Architecture

The configuration of the servers and database for the Test 2 environment will change during the development phase, as the system is performance tuned for the OS and database applications. The final configuration will look very similar to the production configuration below except for the following:

- For the DB partitioning configuration, each node will have 4 CPUs that will be configured with 4 logical partitions / node. Each partition will be using 1 CPUs, so across all 4 nodes there will be 16 partitions.

- All other configuration parameters for the Test 2 environment will be similar to production.

10.4 Production Environment

The Production database server will consist of four IBM pSeries 660 model 6M1 servers with eight 750MHz processors, 16 GB of memory, two Internal 36.4GB disks, and a FastT500 disk array with fifty 73.4 GB hard drives (Since there are four arrays, this is equivalent to approximately 12 TB of storage). Note that the capacity numbers for the production environment may change based on the technical reassessment effort occurring after the detailed design phase of the NSLDS II project.

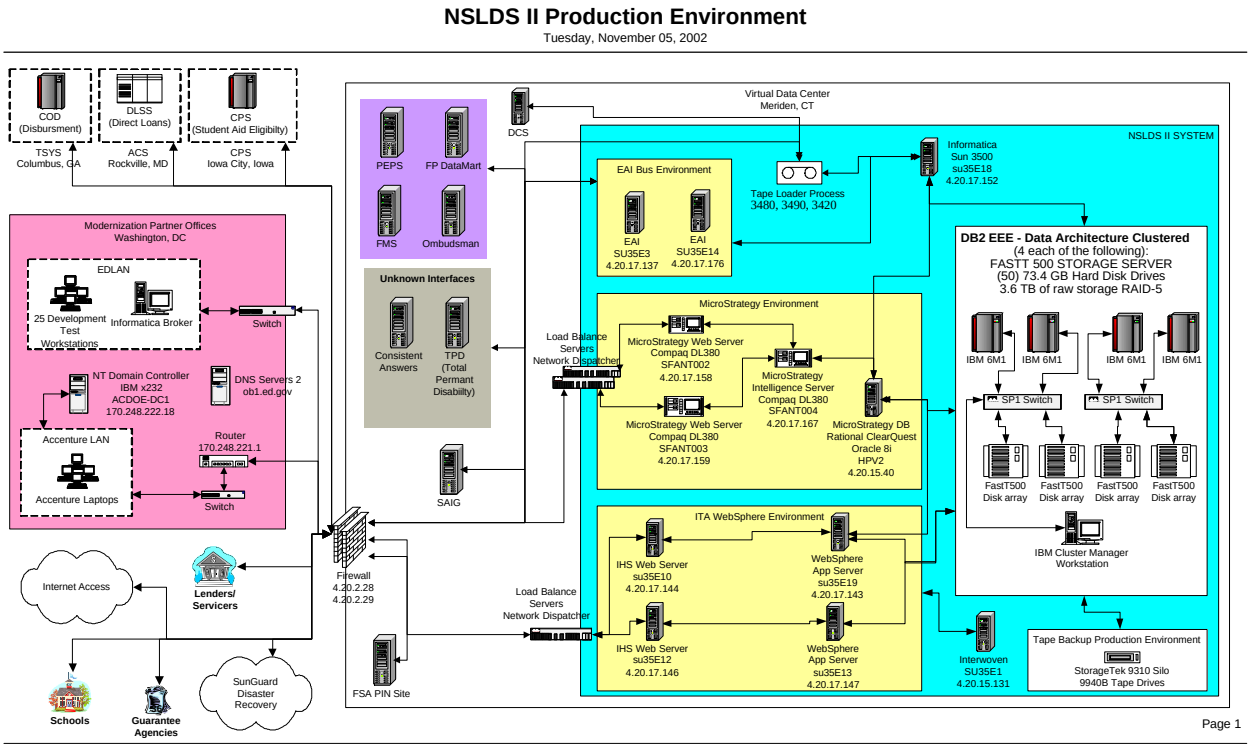


Figure S. NSLDS II Production Environment

The configuration of the servers and database for the production environment will change during the development phase, as the system is performance tuned for the OS and database applications. The following are the expected changes necessary for the Production environment configuration:

- For the DB partitioning configuration, each node will have 8 CPUs that will be configured with 4 logical partitions / node. Each partition will be using 2 CPUs, so across all 4 nodes there we will be 16 partitions.

- For the disk configuration, each disk array will be configured as RAID-5 LUNs. These will be further defined as logical volumes with redundant access paths for controller failover situations. These logical volumes will be used to create file systems where database table spaces will reside.

- There will be a /home/db2inst1 file system, which will be network file system (NFS) mounted across all nodes. This directory will have all the scripts and tools needed for every day administration.

- Table spaces will be created similar to development scheme with subject area tables in their own table spaces.

11 Appendix A - Data Definition Language for EDW and Data Mart

11.1 NSLDS II EDW Table Spreadsheet sorted by Subject Area

11.2 EDW DDL

11.3 Data Mart DDL

12 Appendix B - Data Models for EDW and Data Mart

12.1 EDW Data Model

12.1.1 Default Rate

12.1.2 Direct Loan

12.1.3 Guaranty Agency

12.1.4 Lender

12.1.5 Loan

12.1.6 School

12.1.7 SSCR

12.1.8 Student

12.1.9 Transfer Monitor

12.1.10 System

12.2 Data Mart

12.2.1 CDR

12.2.2 Enrollment

12.2.3 GA and Aggregate Descriptors

12.2.4 Grant

12.2.5 Guaranty Agency

12.2.6 Lender

12.2.7 Loan

12.2.8 Misc Lookups

12.2.9 NSLDS User

12.2.10 On-line Update

12.2.11 Organizational Contact

12.2.12 Prescreening

12.2.13 School

12.2.14 Servicer

12.2.15 SSCR

12.2.16 Student

12.2.17 Transfer Monitor

13 Appendix C - Data Mappings from NSLDS II EDW to Data Mart

Appendix D - AIX Logical Volume Information for Development

14 Appendix E - SQL Commands for Development

15 Appendix F - Capacity Planning for NSLDS II

16 Appendix G - MicroStrategy References

16.1 How to select, edit, create and run a script using MicroStrategy Command Manager 7

16.2 Encryption in MicroStrategy 7

16.3 Encryption in MicroStrategy 7

16.4 Informatica Application Level Security

16.4.1 User and Group Administration in PowerCenter

16.4.2 Creating a Group in PowerCenter

16.4.3 Creating a User in PowerCenter

16.4.4 Managing Privileges in PowerCenter

16.4.5 Available PowerCenter Privileges

16.5 Report and Function Group Associations